

Collection
Ressources Informatiques

HTML 4

Maîtrisez le code source

**Troisième
Edition**

LUC VAN LANCKER

Fichiers à télécharger
RECHERCHER
www.editions-eni.fr

 **INFORMATIQUE TECHNIQUE**



HTML 4

Maîtrisez le code source (3ème édition)

Luc VAN LANCKER



Résumé

Ce livre sur le **HTML 4** s'adresse à toute personne appelée à développer, mettre en place, faire vivre un site web. En effet, pour débiter mais surtout pour progresser dans la conception de sites, il faut inévitablement passer par une bonne compréhension et une bonne maîtrise du **code source des pages**.

Le livre est conçu comme un réel outil de formation, pédagogique de la première à la dernière page, **abondamment illustré** d'exemples et de captures d'écran et constamment à l'affût des éléments réellement pratiques pour le webmestre. Sont ainsi passés en revue le **HTML** (dans sa dernière version et surtout ses derniers usages), les **feuilles de style**, le **JavaScript**, les **CSS**, l'utilisation du **DHTML**, le **XHTML**... Cet ouvrage n'est surtout pas une encyclopédie exhaustive de ces différentes techniques mais un **parcours structuré** de celles-ci. Il fournit aux concepteurs débutants, voire plus confirmés, les règles rigoureuses mais essentielles de la conception professionnelle d'un site Web.

Dans cette nouvelle édition de l'ouvrage, l'auteur s'est attaché à encourager l'élaboration d'un code valide, **respectueux des prescriptions du W3C** et particulièrement de la séparation du contenu (HTML) et de la présentation (feuilles de style CSS). Il recense également les éléments qui peuvent réagir différemment selon le navigateur final (Internet Explorer ou Firefox en l'occurrence.) L'auteur présente aussi quelques outils intéressants (nouvelles versions des validateurs de code, extensions pour Firefox, logiciel de création d'images réactives...).

L'auteur

Dès les débuts d'Internet, Luc **Van Lancker**, enthousiasmé par l'idée de communication universelle que véhiculait ce concept, s'est complètement investi dans ce domaine. C'est un formateur passionné, très au fait des nouvelles technologies liées au web et grand pédagogue.

Retrouvez l'interview de Luc VAN LANCKER : "**Le mot de l'auteur**", à propos de son livre Accessibilité des sites web - Mise en œuvre des directives WCAG 1.0.

Ce livre numérique a été conçu et est diffusé dans le respect des droits d'auteur. Toutes les marques citées ont été déposées par leur éditeur respectif. La loi du 11 Mars 1957 n'autorisant aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les "copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective", et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, "toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayant cause, est illicite" (alinéa 1er de l'article 40). Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles 425 et suivants du Code Pénal. Copyright Editions ENI

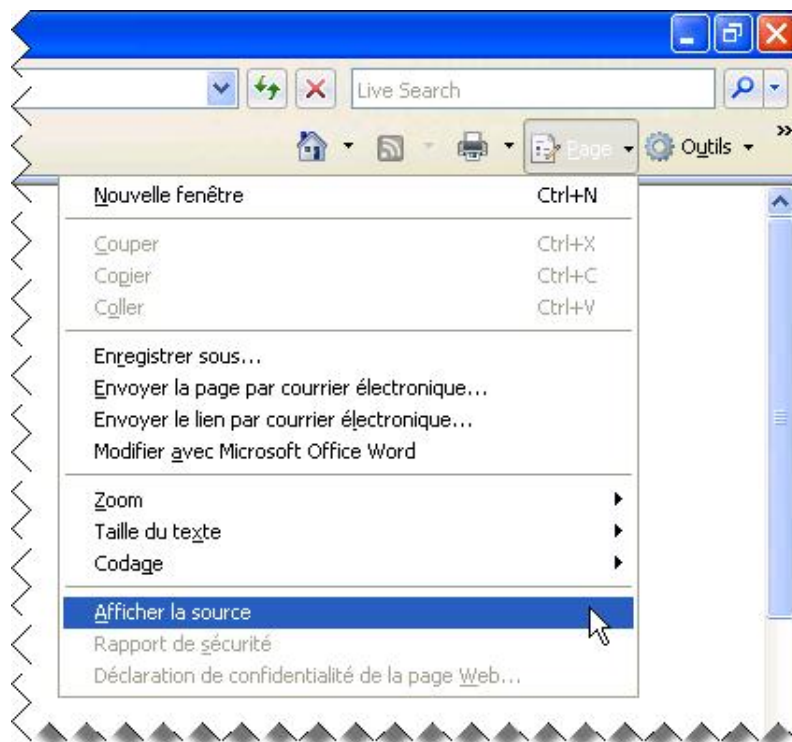
Le Html est le langage du Web

Derrière les pages rencontrées au hasard de votre exploration du Web, se cache leur code source. En effet, le navigateur, Microsoft Internet Explorer ou Firefox (cf. Les navigateurs de notre étude du présent chapitre), ne fait qu'afficher à l'écran la page conçue par l'auteur dans un langage, a priori assez hermétique, qu'est le langage Html. Pourtant, le Html est le langage universel de toutes les pages Web.

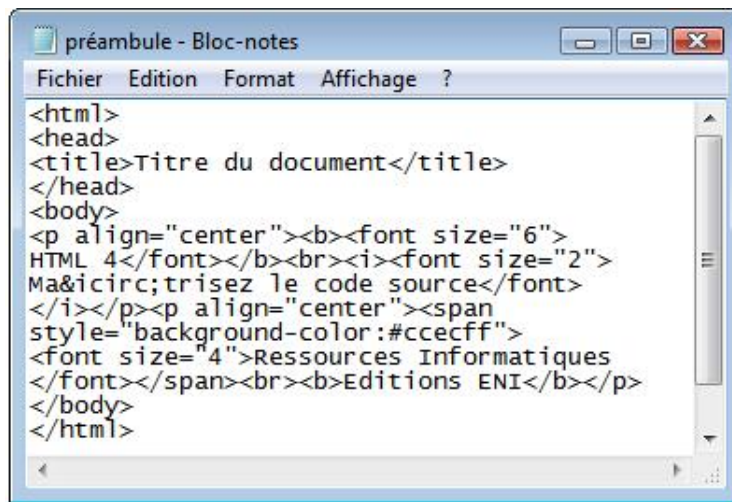


Pour vous en convaincre :

- Dans Internet Explorer 7 : dans **Page**, activez l'option **Afficher la source**.
- Dans Microsoft Internet Explorer 6 : dans le menu **Affichage**, activez l'option **Source**.
- Dans Firefox : dans le menu **Afficher**, activez l'option **Code source de la page**.



Le code source de la page, en langage Html, s'affiche alors dans un "fouillis" de lettres, de chiffres et de caractères cabalistiques, complètement incompréhensible pour les néophytes.



Peut-être avez-vous déjà eu la furieuse envie de créer vous aussi des pages Web pour être un acteur et non plus un spectateur de la toile ?

Au début du Web, cette démarche commençait par l'étude lente et systématique du langage Html avant de pouvoir enfin réaliser, à force de travail et d'heures passées devant son clavier, une première page digne de ce nom.

Aujourd'hui avec des éditeurs Html comme Dreamweaver ou Microsoft Expression (qui rendent transparent l'écriture en langage Html), vous avez peut-être déjà créé des pages dont vous pouvez être légitimement fiers. Cependant, pour évoluer dans le domaine de la publication, l'apprentissage du Html se révèle indispensable.

En effet, malgré des éditeurs Html de plus en plus performants, il vous faudra tôt ou tard "mettre les mains" dans le code source pour les raisons suivantes :

- Les éditeurs Html reprennent uniquement les fonctions du langage Html. L'étude du code source vous permet de mieux connaître les multiples possibilités de votre éditeur Html.
- Les éditeurs Html ne sont pas forcément complets. Certaines balises sont encore ignorées et seront peut-être celles qui vous permettront d'apporter cette touche d'originalité qui fera la différence.
- Les éditeurs Html ne sont pas parfaits. Dans certaines situations, notamment lors d'annulations successives, ils peuvent présenter quelques faiblesses.
- Les éditeurs sont des instruments précieux mais ils ont parfois tendance à écrire pour vous. Avec une connaissance du Html, vous garderez la maîtrise de vos pages.
- Les éditeurs Html ne prennent en considération les feuilles de style (cf. Chapitre Les feuilles de style), aux possibilités si riches, que de façon "très partielle" ; alors que quelques lignes de code suffisent pour transformer une page.
- La plupart des éditeurs Html récents incluent des animations dynamiques ; mais pour apporter des animations un peu originales, il sera encore nécessaire de passer par le code source (cf. Chapitre Les autres langages du web).

Vous aurez également recours au code source :

- Pour ajouter de l'interactivité à vos pages avec quelques éléments de JavaScript (cf. Chapitre Les autres langages du web).
- Pour réaliser des opérations plus complexes, comme la consultation en ligne des bases de données où il conviendra d'inclure directement du code Html dans des lignes de programmation PHP ou ASP.

Le contexte historique

1. Le Html 1.0

L'histoire du Html (*HyperText Markup Language*) commence en 1990. Tim Berners-Lee, informaticien britannique au Centre d'Énergie et de la Recherche Nucléaire de Genève (CERN), veut mettre au point un système pour échanger des informations entre scientifiques par l'intermédiaire d'Internet. Les caractéristiques sont un formatage de texte simple, l'incorporation de graphiques et l'élément qui va s'avérer génial : la possibilité de faire des liens vers d'autres documents.

Le langage Html 1.0 est né. À cause de cet élément hypertexte, tout le projet est baptisé World Wide Web (réseau mondial ou plus communément appelé la toile).

Les éléments induits par ce contexte historique sont :

- À l'origine, les pages Web sont bien d'essence textuelle et bien éloignées de l'aspect visuel qu'on leur a apporté depuis.
- Le Html reprend ses termes de l'anglais qui est la langue la plus répandue dans les milieux scientifiques internationaux.
- À l'époque, le matériel électronique est rustique par rapport au matériel actuel ; disque dur et mémoire de petite capacité, le top de l'affichage est de 256 couleurs. Les lignes téléphoniques sont encore loin de l'ADSL ou du câble.
- Pour s'échanger des documents, on n'utilise aucun logiciel spécifique comme Word, WordPerfect... On a bien créé un langage spécifique et indépendant du système d'exploitation et ce, dans le plus pur esprit d'Internet.

Très rapidement (1992-1993), les premières applications pour visionner en ligne les documents Html (ou navigateurs), font leur apparition : Mosaic, d'abord et Netscape, ensuite. La porte du Web est enfin ouverte au grand public. C'est le "Big-bang" du Web.

Conscientes des énormes possibilités offertes par le langage Html et poussées par l'impatience des pionniers, ces firmes vont très vite créer leurs propres évolutions du code Html, mais de façon quelque peu confuse, voire anarchique.

2. Le Html 2.0

Une version Html 2.0 a bien tenté d'endiguer ce raz-de-marée mais elle ne faisait qu'entériner certaines extensions apportées par les firmes, sans réelle vision pour de futurs développements. La poussée créatrice était bien trop forte et s'est donc poursuivie, surtout que Microsoft a pris le train du Web en marche et a apporté ses propres spécifications. En fait, dès sa parution, le Html 2.0 est déjà, dans la pratique, complètement dépassé.

Conscient de ces dérapages, le même Tim Berners-Lee crée en 1994, le W3C (pour consortium du WWW). Ce consortium reprend des délégués des principaux intervenants du Web et du monde de l'informatique comme par exemple IBM, Microsoft, America Online, Netscape, Apple, Adobe, Macromedia, Sun Microsystems, etc. Les objectifs du W3C sont, à cette époque (et encore aujourd'hui) de standardiser la publication sur le Web et de rendre le Web accessible à tous ses utilisateurs quelles que soient leurs différences de culture, d'éducation, de ressources et/ou de handicap physique.



Ces objectifs ambitieux n'ont trouvé que difficilement un consensus auprès des différents partenaires et la mise en route est laborieuse et peu fructueuse. On en tient pour preuve, une version Html 3.0 qui n'a jamais vu le jour.

3. Le Html 3.2

Il a fallu attendre janvier 1997, pour obtenir une première version réellement standard (avec le Html 3.2) qui a été reconnue et prise en charge par la majorité des navigateurs.

Entre-temps, l'autorité du W3C s'est affirmée et a donné le jour à toute une série de recommandations et de nouveaux langages qui sont maintenant acceptés de façon unanime. Au cours de cet ouvrage, le W3C sera souvent cité comme référence de la publication sur le Web. Le W3C a bien entendu son site www.w3.org qui reprend toutes les spécifications techniques.

4. Le Html 4.0

La dernière version du Html date de décembre 1999. Avec la spécification Html 4.0 qui introduit de nouvelles commandes, officialise les feuilles de style (cf. Chapitre Les feuilles de style) et apporte ses recommandations en terme d'écriture, le langage Html dispose enfin d'un standard qui fait l'unanimité.

Il existe aussi une version supérieure (4.01) qui corrige les erreurs de la version 4.0.

5. Le XHTML 1.0

Internet et le Web sont en perpétuelle évolution. Ainsi, le W3C a déjà posé les bases du successeur du Html 4.0 que nous découvrirons au chapitre Le XHTML, le XHTML 1.0 (*eXtensible HTML*).

Le Html est un langage de balises

1. Les balises

Les balises sont des **commandes** à l'intention du navigateur et saisies entre des signes inférieur à (<) et supérieur à (>).

Ainsi une balise s'écrit <balise>.

En règle générale, à toute balise d'ouverture (ou balise de début) correspond une balise de fermeture (ou balise de fin). Cette balise de fermeture marque la fin de la commande annoncée par la balise d'ouverture. Elle reprend le même énoncé que la balise d'ouverture mais est précédée d'une barre oblique (/). Ainsi à la balise d'ouverture <balise>, correspond la balise de fermeture </balise>.

Par exemple, le texte :

Il est <souligné>important</souligné> d'apprendre le langage <italique>HTML</italique> !

peut se comprendre de la façon suivante :

- écrire "Il est " de façon normale puisque rien n'est spécifié,
- ensuite écrire le mot (et uniquement ce mot) "important" et le souligner,
- reprendre l'écriture normale pour "d'apprendre le langage",
- écrire cette fois-ci en italique le mot "HTML",
- et terminer par "!" en écriture normale.

Ce qui donne la phrase :

Il est **important** d'apprendre le langage *HTML* !

C'est le principe du Html. Chaque fois que l'on donne un ordre (une commande) au navigateur par exemple formater du texte, commencer un tableau ou faire un lien vers une autre page, on met une balise de début. La balise de fermeture signale au navigateur que la commande est terminée.

Ces balises font appel à des termes anglais ou des abréviations de termes anglais qui peuvent les rendre (au premier abord) abstraites et donc complexes, notamment pour les francophones.

Exemples

	b pour bold, ce qui signifie gras.
<i>	i pour italic, ce qui signifie italique.
<p>	p pour paragraph, ce qui signifie paragraphe.
<table>	table, ce qui signifie tableau.
<color>	color, ce qui signifie couleur.
...background...	background, ce qui signifie arrière-plan.
...bgcolor...	bg pour background, ce qui signifie couleur d'arrière-plan.
etc.	

2. Les attributs de la balise

Il est parfois nécessaire de compléter une commande par des spécifications plus précises. Pour ce faire, le langage Html dispose des attributs de la balise.

L'attribut s'insère dans la balise, entre le mot de commande et le signe > final.

La spécification apportée par l'attribut comporte généralement une valeur, celle-ci s'indique en complément de l'attribut par un signe égal (=) suivi de la valeur. Il est recommandé de mettre cette valeur entre guillemets. Le strict respect de la syntaxe veut qu'il n'y ait pas d'espace avant et après le signe égal.

La syntaxe complète d'un attribut est :

`attribut="valeur"`

Il est possible d'utiliser plusieurs attributs, séparés par un espace, dans une même balise.



Pour les amateurs de précision, signalons que le Html reprend la syntaxe du SGML (*Standard Generalized Markup Language*) qui est une norme apparue en 1986 pour la gestion électronique des documents.

Le bon usage des balises

Voici quelques règles simples qu'il est important de respecter lors de l'écriture des balises.

- Les balises Html ne sont pas sensibles aux majuscules et minuscules (insensibles à la casse ou *case insensitive*). Il est, pour le Html, équivalent d'écrire `<BALISE>`, `<Balise>` ou `<balise>`. On a longtemps préconisé d'écrire les balises Html en majuscules pour les différencier du texte normal. Cependant, les évolutions annoncées du Html, comme le langage XHTML, sont, elles, sensibles aux majuscules et minuscules (sensibles à la casse ou *case sensitive*) et l'usage veut que les balises de ces langages soient écrites en minuscules afin d'éviter toute source d'erreur. Pour vous préparer aux futurs développements du Html, il est conseillé de prendre déjà la bonne habitude d'écrire les balises en minuscules.
- La règle générale veut qu'à toute balise d'ouverture `<balise>` soit associée une balise de fermeture `</balise>` ; mais il y a des exceptions :
 - Les exceptions provenant du langage Html lui-même : en effet, certaines balises dites uniques ne comportent pas de balise de fermeture comme par exemple `
`, ``, `<input>`.
 - Les libertés que l'on a prises dans l'écriture du Html à cause du fonctionnement plus ou moins permissif des navigateurs. Ainsi par exemple, la balise de fermeture de paragraphe `</p>` est devenue facultative.

➤ Le XHTML, dernière évolution du Html en date, impose quant à lui une balise (ou signe) de fermeture pour **toutes** les balises.

- Les balises doivent être correctement imbriquées. Lorsqu'on affecte plusieurs balises à un élément, leur ordre de fermeture est essentiel. La première balise de fermeture doit correspondre à la dernière balise d'ouverture non fermée.

Un exemple et tout sera beaucoup plus clair :

Est correct : `<a><c>élément</c>`.

Est incorrect : `<a><c>élément</c>`.

➤ Les navigateurs récents acceptent des balises mal imbriquées, ce que le XHTML n'accepte plus.

- Les valeurs des attributs doivent de manière générale figurer entre des guillemets.

➤ Les navigateurs récents permettent bien souvent l'omission des guillemets. Ici à nouveau, le XHTML nous appelle à plus de rigueur et impose des guillemets pour toutes les valeurs même les valeurs numériques.

Les particularités et pièges du Html

Le langage Html comporte quelques particularités, voire quelques pièges qu'il est bon de connaître avant d'entamer son étude.

- En Html, un seul espace est pris en compte par le navigateur. Ainsi si vous encodez :

`gauche droite`

à l'affichage le texte apparaîtra `gauche droite` avec un seul espace.

Dans le même ordre d'idée, les tabulations de votre éditeur de texte ne seront pas prises en compte. Les espaces supplémentaires sont introduits par l'espace insécable qui se note ` `.

- En Html, les passages à la ligne de l'éditeur de texte ne sont pas pris en compte.

Si vous encodez :

`gauche`

`droite`

À l'affichage, vous aurez `gauche droite`. Le passage à la ligne de l'éditeur est interprété comme un espace. Une balise Html particulière existe pour forcer le passage à la ligne.

- Le format de caractères adopté pour le Html est le format ASCII qui permet de représenter seulement 128 caractères soit l'alphabet simple, les chiffres et quelques caractères propres à l'informatique. Ce format présente l'avantage d'être très compact et était donc idéal à l'époque, compte tenu du matériel informatique et du débit des lignes téléphoniques. Ce format simpliste ne comporte pas d'accents, ce qui avec l'anglais utilisé par la communauté scientifique de l'époque, ne constituait pas un réel problème. Avec l'ouverture du Web, pour les langues utilisant des accents comme le français, l'allemand ou l'espagnol, il a fallu introduire des caractères spéciaux (cf. chapitre Le texte et sa présentation - Les caractères spéciaux). Ces accents, bien que dûment encodés dans les éditeurs de texte, ne sont pas interprétés par les navigateurs. Ainsi il faut écrire `"C'est l'été"` pour obtenir "C'est l'été".
- Dans le même ordre d'idée, certains caractères propres au Html comme `<`, `>`, `&` et `"` sont encodés de façon spéciale (voir Annexe Les caractères spéciaux).
- Le Html n'est pas une science exacte. Le Html doit passer par une application qui affiche le code source à l'écran. C'est le même logiciel, que l'on appelle navigateur, qui vous permet de surfer sur le Net et de voir sur votre écran les "pages" qu'il a interceptées. Il y a, hélas, beaucoup de marques et de types de navigateurs différents. Les plus connus sont Internet Explorer maintenant dans ses versions 6 et 7 ainsi que Firefox 2 ou 3 mais il en existe beaucoup d'autres comme, par exemple Opera ou Safari. Chaque navigateur a dans certaines circonstances, sa propre façon d'interpréter le code source. Dans certaines situations (rares heureusement), en fonction du navigateur que vous utilisez sur votre ordinateur ou du navigateur que l'utilisateur final utilisera, le code source peut être rendu différemment. À titre d'exemple ou d'anecdote, il y avait de petites différences d'affichage du code source entre la version 5.5 de Microsoft Internet Explorer sous Windows et la même version du même navigateur de la même marque sous Macintosh.
- En HTML, vous n'avez pas la maîtrise totale de votre document. À la différence de votre traitement de texte préféré qui restitue exactement votre document sur une feuille de papier avec votre police de caractères et votre mise en page, vous ne saurez jamais exactement ce que le navigateur de votre lecteur du bout du monde affichera sur son écran. Celui-ci a peut-être une autre résolution d'écran, une autre police par défaut, une autre résolution graphique, etc.

Le document Html minimum

Un fichier Html comprend en principe deux parties :

- L'en-tête (le "header" en anglais) qui contient des données générales relatives au document, comme le nom de l'auteur, un bref résumé du document, sa date de création, etc. La donnée la plus importante, voire indispensable, est le titre du document (nous reviendrons plus en détails sur ces données au chapitre L'en-tête d'un document HTML).
- Le corps du document (le "body" en anglais) qui contient le texte proprement dit du document et qui sera affiché par le navigateur.

Le document Html minimum aura la forme :

```
<html>
```

Signale au navigateur le début d'un document Html.

```
<head>
```

Marque le début de la zone d'en-tête.

```
<title>Titre du document</title>
```

C'est le seul élément visible de l'en-tête. C'est le titre du document qui vient s'afficher dans la barre de titre du navigateur. Trop souvent négligé par les débutants, ce titre indique le contenu de la présentation de la page dans le navigateur.



```
</head>
```

Marque la fin de la zone d'en-tête.

```
<body>
```

Début du corps du document.

... Le contenu du document ...

Tout ce qui est compris entre les balises `<body>` et `</body>` sera affiché dans la fenêtre du navigateur.



</body>

Fin du corps du document.

</html>

Signale au navigateur la fin du document Html.

La façon de procéder

Aucun langage de programmation, aussi simple soit-il, ne s'apprend par la lecture d'un ouvrage. C'est par la pratique, en reproduisant soi-même les exemples, que l'on peut en comprendre et en maîtriser le fonctionnement. Il est donc vivement conseillé de reprendre les exemples qui vont suivre.

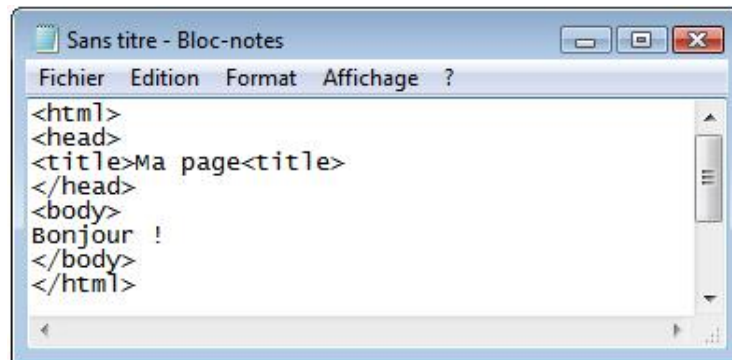
Pour écrire du code Html, on utilise un éditeur de texte qui se retrouve dans tous les systèmes d'exploitation. C'est le cas du Bloc-notes, aussi appelé Notepad sur Windows ou SimpleText sur Macintosh.

Dans le cas de Windows, pour accéder à cette petite application, dont certains ignorent même parfois l'existence, dans le menu **Démarrer - Programmes - Accessoires**, activez l'option **Bloc-notes**.

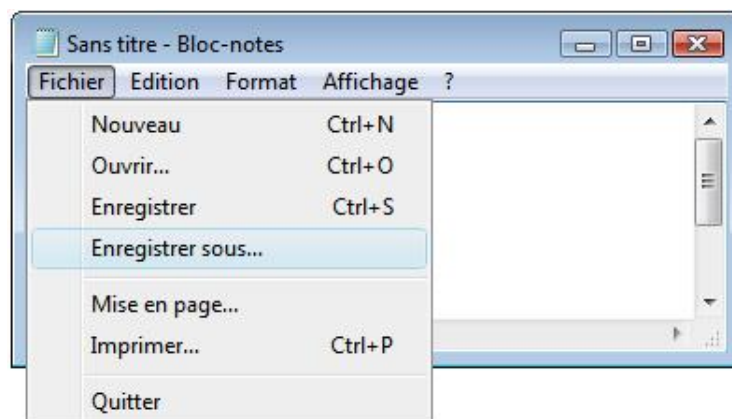
Cet éditeur de texte n'a pas beaucoup de fonctions par rapport à votre traitement de texte préféré : il ne peut pas mettre en gras, italique ou souligné, ni effectuer des alignements ou insérer des images ou des tableaux, mais il est "parfait" pour ce qu'on souhaite faire, soit encoder du texte pur (ASCII).

Encodez le document Html minimum dans le Bloc-notes. Soit :

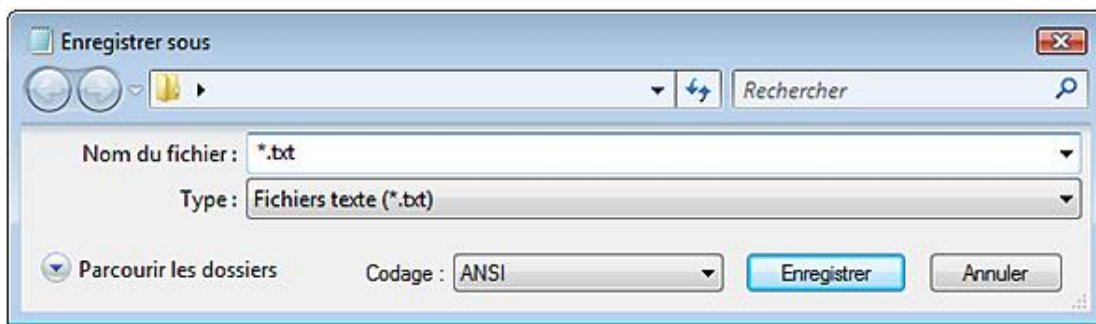
```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Bonjour !
</body>
</html>
```



- Dans le menu **Fichier**, activez l'option **Enregistrer sous...**



Après avoir déterminé l'emplacement du fichier, une boîte de dialogue apparaît comme suit :



- Dans la zone **Type**, activez l'option **Tous les fichiers**. Le type de document n'est pas un fichier texte (*.txt), comme présenté par défaut.

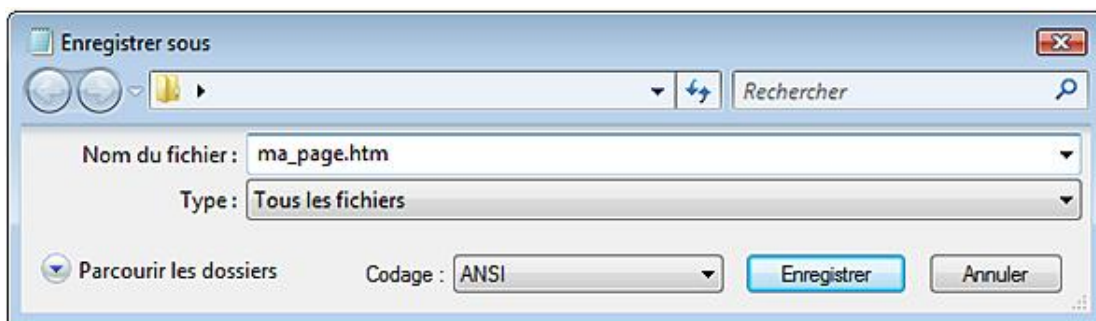


- Donnez un nom au fichier. Les documents Html portent l'extension *.htm ou *.html. Un bon conseil est d'adopter votre extension et pourquoi pas celle de la majorité des éditeurs Html qui est *.htm.

Pour le nom, prenez l'habitude d'encoder le nom entièrement en minuscules, sans majuscule à la première lettre. Cela n'a rien d'obligatoire et même si Windows s'en accomode très bien, il y a d'autres systèmes d'exploitation (et cela sera peut-être celui de l'ordinateur qui hébergera votre site) qui sont beaucoup plus pointilleux sur les majuscules et minuscules (case sensitive) comme Unix. On évite ainsi une première source d'erreur possible.

En outre, Windows de Microsoft vous permet (et vous a donc habitué) de mettre des espaces dans le nom de vos fichiers. Cette permissivité n'est pas acceptable sous d'autres systèmes d'exploitation, ni même sous d'autres navigateurs que Microsoft Internet Explorer. Les espaces sont à proscrire définitivement du nom de tous vos fichiers (pages, image, etc.) relatifs à la publication sur le Web !

Un nom correct pour ce fichier peut être ma_page.htm.



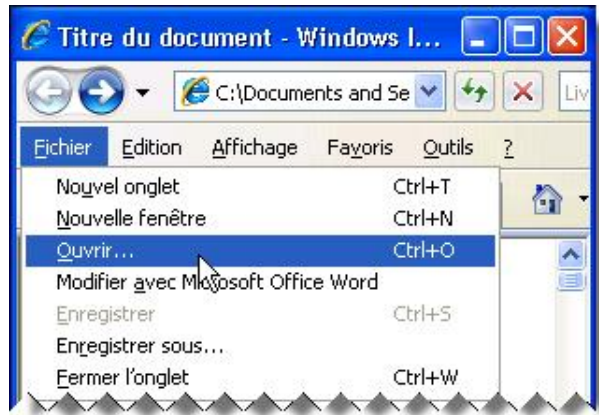
- Cliquez sur le bouton **Enregistrer**.
- Réduisez ou minimisez (plutôt que de fermer) l'application Bloc-notes.



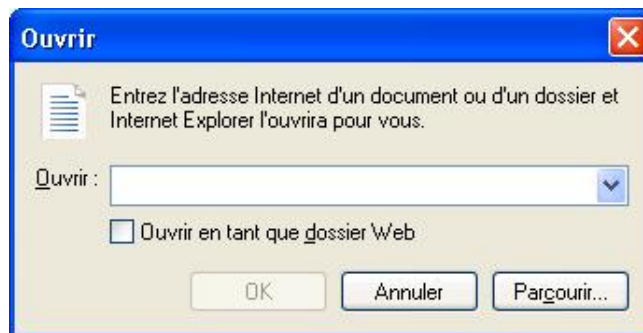
- Pour afficher votre page, vous devez ouvrir votre navigateur.

Comme la création de pages Web se réalise hors connexion et sur votre ordinateur, il faut indiquer le chemin vers le fichier à afficher.

- Dans le menu **Fichier**, activez l'option **Ouvrir**.

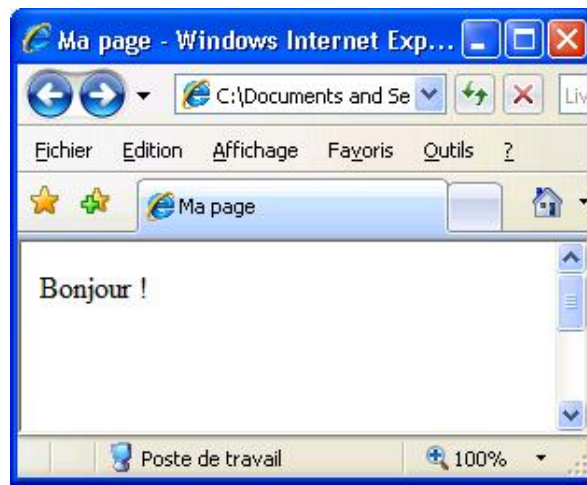


La fenêtre suivante apparaît :



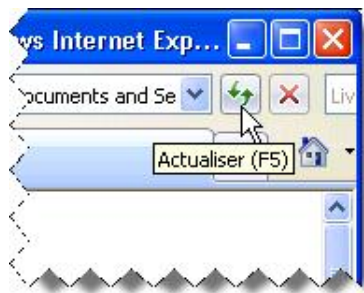
- Cliquez sur le bouton **Parcourir** et indiquez l'emplacement où est enregistré votre document puis cliquez sur le bouton **OK**.

Votre premier document écrit en Html s'affiche alors dans le navigateur.



- Cliquez sur l'icône **Réduire** de votre fenêtre pour la minimiser plutôt que la réduire.

➤ Le travail en code source Html sera un "ballet" entre les modifications dans le Bloc-notes et l'affichage dans le navigateur. N'oubliez pas d'actualiser pour que le navigateur puisse tenir compte de vos modifications.



➤ "Mais tout le monde peut consulter le code source de mes pages !".

Cette remarque fait partie des grands classiques de toute formation au langage source qu'est le Html. Mais :

- Cet état de fait correspond entièrement à l'esprit de partage de l'information et des connaissances qui a fait (et fera encore sûrement longtemps) l'essence d'Internet.
- Il faut admettre que depuis le temps, sur les centaines de millions de pages Web, les multiples possibilités et astuces du langage Html ont vraisemblablement déjà été explorées.

Les navigateurs de notre étude

Dans le panorama des navigateurs disponibles (courant 2008), Microsoft Internet Explorer 6 et 7 maintiennent une position prépondérante mais Firefox grignote, de jour en jour, des parts importantes de marché et se révèle le seul concurrent notoire d'Internet Explorer.

Notre étude sera donc centrée sur Internet Explorer 7 et Firefox 3. Nous ne manquerons pas de mettre en évidence les différences d'interprétation des balises Html et des feuilles de style CSS entre ces deux navigateurs.

Les versions 6 d'Internet Explorer et 2 de Firefox, encore bien présentes dans le parc informatique ne seront néanmoins pas oubliés.

1. Internet Explorer 7

Internet Explorer 7, le navigateur de Microsoft n'est plus à présenter avec la généralisation progressive de Windows Vista avec lequel il s'installe par défaut. Il existe également une version téléchargeable pour Windows XP SP2.

 chap1_Explorer%207.png

Malgré une interface graphique plus fonctionnelle avec la navigation par onglets, le moteur d'Internet Explorer 7 est toujours basé sur celui d'Internet Explorer 4 remontant à octobre 1997. Les webmasters confirmés lui reprochent d'être peu scrupuleux des normes du W3C, spécialement en ce qui concerne les feuilles de style CSS.

Microsoft promet que la version 8 d'Internet Explorer, prévue pour 2009, sera [enfin] respectueuse des directives du W3C.

2. Firefox

Firefox est un navigateur Web libre, développé et distribué par la Fondation Mozilla. Le nom Firefox (littéralement *renard de feu*) est le nom anglais d'un animal appelé panda rouge ou petit panda (*Ailurus fulgens*) en français.



Apparu officiellement en novembre 2004, il connaît directement un succès évident. Pour preuve, moins d'un an après sa sortie officielle, son nombre de téléchargements atteint les 100 millions. Ses parts de marché peuvent être estimées actuellement à près de 40 % et le positionnent ainsi en concurrent d'Internet Explorer.

La fondation Mozilla est née après la libéralisation du code et la déconfiture du navigateur Netscape, qui avait été le leader incontesté du marché des navigateurs jusqu'en 1997. Ainsi, Firefox peut être considéré comme ayant quelques liens de parenté avec Netscape dont il a repris, par exemple, le moteur Gecko qui gère le rendu des pages.

Le projet Mozilla est né d'une communauté de développement bénévole, qui n'est pas sans rappeler de façon sympathique les premières années d'Internet et du Web. Cette communauté, ouverte à toutes formes de partenariat, est aussi garante d'un développement continu.

Le projet Mozilla vise explicitement le respect des standards du Web, tels qu'ils sont promulgués par le W3C, en renonçant à vouloir développer des technologies propriétaires ; ce qui ne peut que réjouir les webmasters.

Une petite anecdote pour terminer. Si vous encodez `about:Mozilla` dans la barre d'adresse de Firefox 3, le texte suivant apparaît.

"Mammon s'était endormi. Et la bête réincarnée se répandit sur la terre et son nombre se fit légion. Et ils parlèrent au Temps et ils firent l'offrande de leur moisson au feu, avec la ruse des renards. Et ils bâtirent un nouveau monde à leur

image comme le promettaient les paroles sacrées, et ils parlèrent de la bête avec leurs enfants. Lorsque Mammon se réveilla, voilà ! ce n'était plus rien qu'un disciple (d'après Le Livre de Mozilla)".

Dans ce texte, les métaphores sont nombreuses. Relevons Mammon pour Internet Explorer, la bête réincarnée pour Netscape et le projet Mozilla et les paroles sacrées pour les recommandations du W3C.

Les feuilles de style

Voici une sélection des feuilles de style les plus utilisées.

1. Les feuilles de style de police (font)

font-family

- définit un nom de police ou une famille de police.
- nom de police précise (Arial, Times, Helvetica...) ou famille (serif, sans-serif, cursive, fantasy, monospace).
- `h3 {font-family:Arial}`

font-size

- définit la taille de la police.
- xx-small ou x-small ou small ou médium ou large ou x-large ou xx-large. ou larger ou smaller ou taille précise en points (pt), inches (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `p {font-size:72pt}`

font-style

- définit le style de l'écriture.
- normal ou italic (pour italique) ou oblique.
- `h3 {font-style:italic}`

font-variant

- écrit de façon normale ou en petites majuscules.
- normal ou small-caps.
- `p {font-variant:small-caps}`

font-weight

- définit l'épaisseur de la police ou du caractère.
- normal ou bold (gras) ou bolder ou lighter ou valeur numérique soit (100|200|300|400|500|600|700|800|900).
- `p {font-weight:bold}`

font

- raccourci pour les différentes propriétés de police.
- exemple : ``

Exemple

```

<html>
<head>
<title>CSS</title>
</head>
<body>
<h2 style="font-family:cursive">Les styles de police</h2>
Ils modifient la <span style="font-size:36px">taille</span>.<br>
Je peux écrire en <span style="font-style:italic">italique</span><br>
en <span style="font-weight:900">gras</span> et en
<span style="font-variant:small-caps">majuscules</span>.
</body>
</html>

```

Résultat



2. Les feuilles de style de texte (text)

text-align

- définit l'alignement du texte à gauche, au centre, à droite ou justifié.
- left ou center ou right ou justify.
- `h1 {text-align:center}`

text-indent

- définit un retrait dans la première ligne d'un bloc de texte.
- spécifié en inches (in) ou en centimètres (cm) ou en pixels (px).
- `p {text-indent:1cm}`

text-decoration

- définit une "décoration" du texte, soit souligné, barré, surligné, clignotant ou sans.
- underline ou line-through ou overline ou blink ou none.
- `a:visited {text-decoration:blink}`

text-transform

- définit la casse du texte (majuscule, minuscule).
- uppercase (met les caractères en majuscules) ou lowercase (met les caractères en minuscules) ou capitalize (met le premier caractère en majuscule).
- `p {text-transform:uppercase}`

line-height

- définit l'interligne, soit l'espace entre les lignes du texte.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `p {line-height:10pt}`

letter-spacing

- définit l'espace entre les lettres.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `p {letter-spacing:2pt}`

color

- définit la couleur du texte.
- par exemple en hexadécimal.
- `h3 {color:#000080}`

width

- détermine la longueur d'un élément.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `h1 {width:200px}`

height

- détermine la hauteur d'un élément.
- en points (pt), inches (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `h1 {height:100px}`

Exemple

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<h1 style="text-align:center;
```

```

        text-transform:capitalize;
        letter-spacing:25px;">style</h1>
<p style="width:225px;
    text-align:justify;
    text-indent:1cm; color:red;
    line-height:11px">
Les feuilles de style permettent une présentation de texte nettement
plus riche et plus variée qu'en Html.
</p>
</body>
</html>

```

Résultat



3. Les feuilles de style d'arrière-plan (background)

background-color

- définit la couleur de l'arrière-plan.
- couleur (par exemple en hexadécimal) ou transparent.
- `h1 {background-color:#000000}`

background-image

- définit l'image de l'arrière-plan.
- `url(fichier_image)` soit l'adresse (url) de l'image.
- `body {background-image:url(image.gif)}`

background-repeat

- définit la façon de répéter l'image d'arrière-plan.
- `repeat` ou `no-repeat` ou `repeat-x` (répétitions horizontales) ou `repeat-y` (répétitions verticales).
- `p {background-image:url(image.gif); background-repeat:repeat-4}`

background-attachment

- spécifie si l'image d'arrière-plan reste fixe avec les déplacements de l'écran.

- scroll ou fixed.

- `body {background-image:url(image.gif); background-attachement: fixed}`

background-position

- spécifie la position de l'image d'arrière-plan par rapport au coin supérieur gauche de la fenêtre.
- verticalement : top ou center ou bottom ou en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- horizontalement : left ou center ou right ou en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `body {background-image:url(img.gif); background-position: right top}`

background

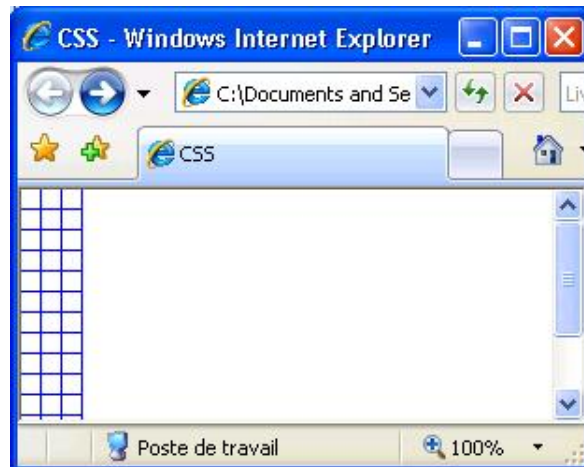
- raccourci pour les différentes propriétés d'arrière-plan.
- `body {background:url(image.gif) fixed repeat}`

Exemple

Mettez une image à l'arrière-plan qui sera répétée verticalement.

```
<html>
<head>
<title>CSS</title>
</head>
<body style="background-image:url(cahier.gif);
           background-repeat:repeat-y">
</body>
</html>
```

Résultat



4. Les feuilles de style de marges ou de retraits (margin)

margin-top

- détermine la valeur de la marge supérieure.

- en unité de longueur ou auto.

- `img {margin-top:5px}`

`margin-right`

- détermine la valeur de la marge droite.

- en unité de longueur ou auto.

- `img {margin-right:5px}`

`margin-bottom`

- détermine la valeur de la marge inférieure.

- en unité de longueur ou auto.

- `img{margin-bottom:5px}`

`margin-left`

- détermine la valeur de la marge gauche.

- en unité de longueur ou auto.

- `h3 {margin-left:5px}`

`margin`

- regroupe les différentes propriétés de la marge.

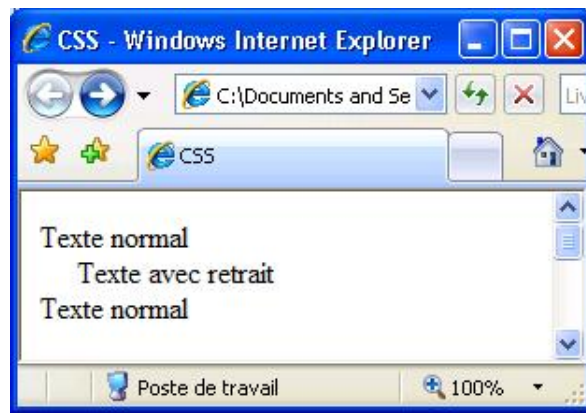
- `img {margin 5px}`

Exemple

Effectuez un retrait du texte avec une feuille de style.

```
<html>
<head>
<title>CSS</title>
</head>
<body>
Texte normal<br>
<span style="margin-left:20px">Texte avec retrait</span><br>
Texte normal<br>
</body>
</html>
```

Résultat



5. Les feuilles de style de bordure (border)

`border-top-width`

- donne l'épaisseur du bord supérieur.
- thin (fin) ou medium ou thick (gros) ou spécifié par l'auteur en pixels.
- `table{border-top-width:thin}`

`border-right-width`

- donne l'épaisseur du bord droit.
- thin ou medium ou thick ou spécifié par l'auteur en pixels.
- `table {border-right-width:medium}`

`border-bottom-width`

- donne l'épaisseur du bord inférieur.
- thin ou medium ou thick ou spécifié par l'auteur en pixels.
- `table{border-bottom-width:thick}`

`border-left-width`

- donne l'épaisseur du bord gauche.
- thin ou medium ou thick ou spécifié par l'auteur en pixels.
- `table {border-left-width:0.5cm}`

`border-width`

- regroupe les différentes propriétés de border-width.
- `table {border-width:thin}`

`border-color`

- détermine la couleur de la bordure.
- se déclina aussi en border-top/right/bottom/left-color.
- `table {border-color:yellow}`

border-style

- détermine le style du trait de la bordure.
- se déclina aussi en border-top/right/bottom/left-style.
- none ou solid ou double ou groove ou ridge ou inset ou outset.
- `table {border-style:none}`

border

- regroupe toutes les propriétés des bords.
- `table {border:0.2cm groove orange}`

Exemple

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<div style=" width: 200px;
            text-align: center;
            border-top-width: 3px;
            border-right-width: 25px;
            border-bottom-width: 3px;
            border-left-width: 25px;
            border-color: red;
            border-style: solid" >
... Votre texte ...
</div>
</body>
</html>
```

Résultat



6. Les feuilles de style d'enrobage (padding)

padding-top

- valeur de remplissage haut entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-top:3px}`

padding-right

- valeur de remplissage droit entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-right: 3px}`

padding-bottom

- valeur de remplissage bas entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-bottom: 3px}`

padding-left

- valeur de remplissage gauche entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-left: 3px}`

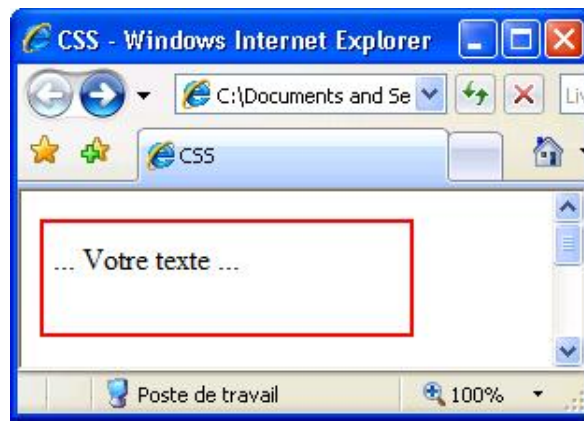
padding

- regroupe les différentes propriétés de remplissage.
- `td {padding:3mm 8mm}`

Exemple (marges intérieures d'une cellule de tableau)

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<div style="width:200px;
        padding-top:10px;
        padding-right:5px;
        padding-bottom:30px;
        padding-left:5px;
        border-color:red;
        border-width:thin;
        border-style:solid">
... Votre texte ...
</div>
</body>
</html>
```

Résultat



7. Les feuilles de style pour les listes

list-style-type

- détermine le type de puce ou de numérotation.
- disc (cercle plein) ou circle (cercle vide) ou square (carré).
- decimal ou lower-roman ou upper-roman ou lower-alpha ou upper-alpha.
- `ul {list-style-type:square}`

list-style-image

- permet de remplacer les puces par une image.
- `url(fichier_image)` ou `none`.
- `ul {list-style-image:url(image.gif)}`

list-style-position

- spécifie si les puces sont à l'intérieur ou à l'extérieur du texte.
- `inside` ou `outside`.
- `ul {list-style-position:inside}`

list-style

- regroupe toutes les propriétés de liste.
- `ul {list-style: outside url(dot.gif)}`

Exemple

Voici une liste à puces avec une image (sans passer par un tableau !).

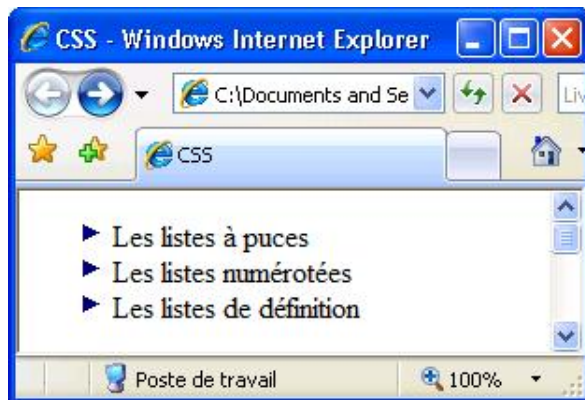
```
<html>
<head>
<title>CSS</title>
</head>
<body>
<ul style=" list-style-image:url(image.gif)">
```

```

<li>les listes à puces</li>
<li>les listes numérotées</li>
<li>Les listes de définition</li>
</ul>
</body>
</html>

```

Résultat



8. Les feuilles de style diverses

page-break-after

- ajoute un saut de page après l'élément. L'impression de la suite se fera sur une nouvelle page.
- always (toujours) ou auto.
- `p {page-break-after: always}`

page-break-before

- ajoute un saut de page avant l'élément. L'impression de la suite se fera sur une nouvelle page.
- always (toujours) ou auto.
- `p {page-break-after: always}`

visibility

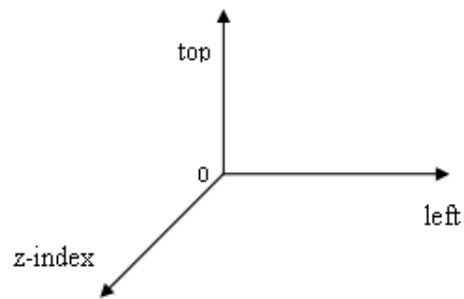
- rend un élément visible ou caché.
- visible ou hidden.

z-index

- permet d'attribuer en ordre sur l'axe vertical. La position 0 est la plus en arrière. Les valeurs supérieures à 0 déterminent l'ordre. Ainsi 0 est plus en arrière que 1 et 1 est plus en arrière que 2...
- ``
- `<div style="position:absolute; top:100; left:100; z-index:1">Votre texte</div>`

Illustrons ce dernier exemple :

La propriété z-index permet d'ordonner des éléments sur l'axe vertical.



```
<html>
<head>
<title>CSS</title>
</head>
<body>

<div style="color:red; position:absolute; top:27; left:100;
z-index:1"><h1>GSM</h1></div>
</body>
</html>
```

L'image gsm.gif a une position 4 sur l'axe z et sera donc placée devant le texte GSM du <div>.

Résultat



9. Les pseudo-classes

a:link

- permet d'attribuer des feuilles de style au lien (non-visité).
- `a:link {text-decoration:none}`

a:hover

- permet d'attribuer des feuilles de style lors du survol du lien.
- `a:hover{color:red; text-transform:uppercase}`

a:active

- permet d'attribuer des feuilles de style au lien actif.
- `a:active{font-weight:bold; color:purple}`

a:visited

- permet d'attribuer des feuilles de style au lien visité.
- a:visited {color: blue}

cursor

- permet de changer la forme du curseur.
- auto ou crosshair (croix) ou default ou hand (main) ou move (déplacement) ou text (un texte éditable généralement I) x-resize (direction n, s, e, w) ou wait (sablier) ou help (point d'interrogation).
- p{cursor:wait}

Exemple

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<h1 style="cursor:help">Surprise ...</h1>
</body>
</html>
```

Résultat



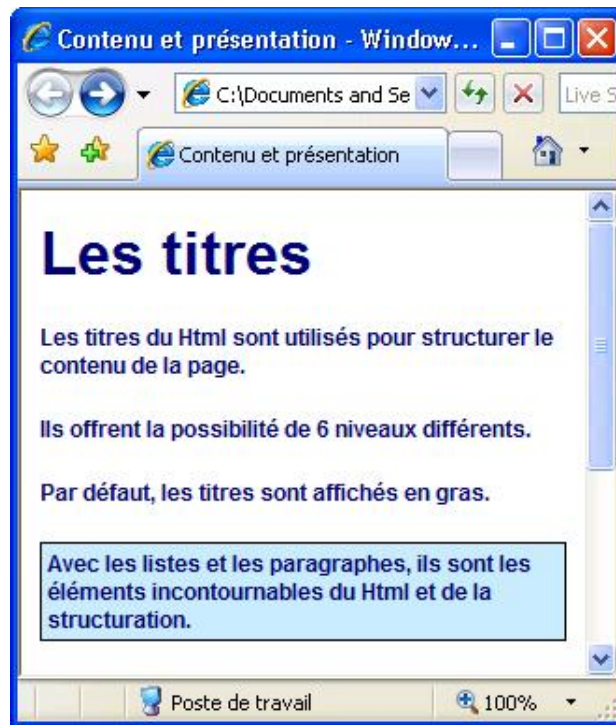
Les autres formes possibles (sous Windows) sont :

crosshair	+
hand	☞
move	↕
text	I
pointer	☞
resize	➡
wait	⌚

Séparer le contenu de la présentation

Les feuilles de style sont surtout utilisées dans l'écriture actuelle des pages Html pour séparer le contenu de tous les effets de présentation.

Découvrons cette façon d'écrire et ses avantages par un exemple.



1. Écriture déclassée

```
<html>
<head>
<title>Contenu et présentation</title>
</head>
<body>
<p><b><font face="Arial"><font size="6"><font color="#000080">
Les titres</font></font></font></b></p>
<p><b><font face="Arial"><font size="2"><font color="#000080">
Les titres du Html sont utilisés pour structurer le contenu de la page.
</font></font></font></b></p>
<p><b><font face="Arial"><font size="2"><font color="#000080">
Ils offrent la possibilité de 6 niveaux différents.</font></font></font>
</b></p>
<p><i><b><font face="Arial"><font size="2"><font color="#000080">
Par défaut, les titres sont affichés en gras.</font></font></font></b>
</i></p>
<table border="1" width="100%" cellpadding="5" bordercolor="black"
bgcolor="#ccecff" cellspacing="0">
<tr>
<td>
<p><b><font face="Arial"><font size="2"><font color="#000080">
Avec les listes et les paragraphes, ils sont les éléments
incontournables du Html et de la structuration.</font></font></font></b>
</td>
</tr>
</table>
</body>
</html>
```

Commentaires

- Pour les quelques lignes que comporte la page, il y a beaucoup d'éléments de code.
- Avec la présentation mélangée au contenu, le code source est un véritable fouillis de balises.
- Le code est difficilement lisible, même pour les initiés.
- Le contenu textuel de la page est noyé dans les éléments de code.
- Le code comporte de nombreux éléments redondants.
- Pour réaliser les effets graphiques, il faut passer par des astuces. Ici un tableau.
- Le document n'est pas un document Html 4.0 valide.

2. Écriture préconisée

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Contenu et présentation</title>
<style type="text/css">
h1 {font-family: Arial, sans-serif;
    color: navy;}
p { font-family: Arial, sans-serif;
    font-size: 12px;
    font-weight: bold;
    color: navy;}
div {font-family: Arial, sans-serif;
    font-size: 12px;
    font weight: bold;
    padding: 3px;
    color: navy;
    border: solid 1px black;
    background-color: #ccecff;}
</style>
</head>
<body>
<h1>Les titres</h1>
<p>Les titres du Html sont utilisés pour structurer le contenu de la
page.</p>
<p>Ils offrent la possibilité de 6 niveaux différents.</p>
<p>Par défaut, les titres sont affichés en gras.</p>
<div>Avec les listes et les paragraphes, ils sont les éléments
incontournables du Html et de la structuration.</div>
</body>
</html>
```

Commentaires

- Le contenu textuel est clairement identifiable. Il est reproduit simplement entouré d'une balise d'ouverture et de fermeture.
- Le contenu est clairement structuré par les balises de titre et de paragraphe.
- Tous les éléments de présentation sont repris en charge par des feuilles de style.
- Ces feuilles de style sont codées à un emplacement distinct du contenu.

- Après un apprentissage, les effets de présentation des éléments de style CSS sont aisément lisibles.
- Le code Html, ainsi débarrassé des balises de présentation, est plus compatible entre les différents navigateurs.

Les grands classiques des feuilles de style

Les feuilles de style permettent en quelques lignes de code de transformer visuellement votre page ou votre site. En voici une sélection !

1. Enlever le soulignement des liens

Pour enlever le soulignement des liens de toute la page, ajoutez entre les balises `<head> ... </head>` :

```
<style type="text/css">
a{text-decoration:none}
</style>
```

Exemple

```
<html>
<head>
<style>
<style type="text/css">
a {text-decoration:none}
</style>
</head>
<body>
<a href="test.htm">Ceci est bien un lien</a>
</body>
</html>
```

Résultat



2. Changer l'apparence d'un lien au survol de la souris

a. Mettre le lien en gras

Ajoutez entre les balises `<head> ... </head>`.

```
<style type="text/css">
a:hover{font-weight:bold;}
</style>
```

b. Mettre le lien en italique

Ajoutez entre les balises `<head> ... </head>`.

```
<style type="text/css">
a:hover{font-style:italic;}
</style>
```

```
</style>
```

c. Changer la couleur du lien

Ajoutez entre les balises <head> ... </head>.

```
<style type="text/css">
a:hover{color:red;}
</style>
```

d. Changer la taille de la police du lien

Ajoutez entre les balises <head> ... </head>.

```
<style type="text/css">
a:hover{font-size:25px;}
</style>
```

e. Ajouter un arrière-plan

Ajoutez entre les balises <head> ... </head>.

```
<style type="text/css">
a:hover{background-color:lime;}
</style>
```

f. Appliquer un soulignement

Ajoutez entre les balises <head> ... </head>.

```
<style type="text/css">
a:hover{text-decoration:underline;}
</style>
```

3. Effectuer des liens de couleur différente sur une même page

Ajoutez entre les balises <head> ... </head>.

```
<style type="text/css">
a.lien1 {color: blue; text-decoration: underline;}
a.lien2 {color: green; text-decoration: underline;}
</style>
```

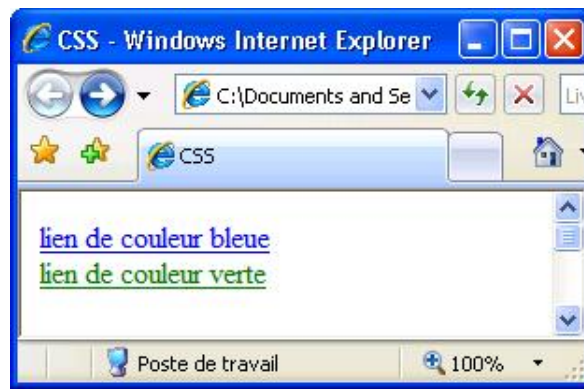
Vous avez ainsi créé deux classes : lien1 de couleur bleue et lien2 de couleur verte.

Dans le <body>, il suffit selon le lien d'appeler la classe lien1 ou lien2.

Exemple

```
<body>
<a class="lien1" href="destination">lien de couleur bleue</a><br>
<a class="lien2" href="destination">lien de couleur verte</a>
</body>
```

Résultat



4. Ajouter une info-bulle sur un mot

Ceci est possible très simplement avec la balise ``.

Exemple

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<span title="Texte de l'infobulle">Texte</span>
</body>
</html>
```

Résultat



5. Égayer vos formulaires

Les feuilles de style permettent également d'égayer des formulaires.

- Faites une ligne de texte avec un arrière-plan (background) d'un bleu léger, avec les lettres de couleur bleue (blue) et en caractères gras (bold), le tout avec un bord de 1 pixel en bleu.

```
<body>
<form action="">
<input type="text" size="30" value="Les feuilles de style"
style="background-color: #elfeff; font-weight: bold; color:
blue;border: 1px solid blue;">
</form>
</body>
```

- Affectez à une zone de texte un arrière-plan jaune pâle et un double bord vert.


```

<body>
<form action="">
<textarea rows="3" cols="26" style="background-color: lightyellow;
border: 5px double green">
</textarea>
</form>
</form>

```

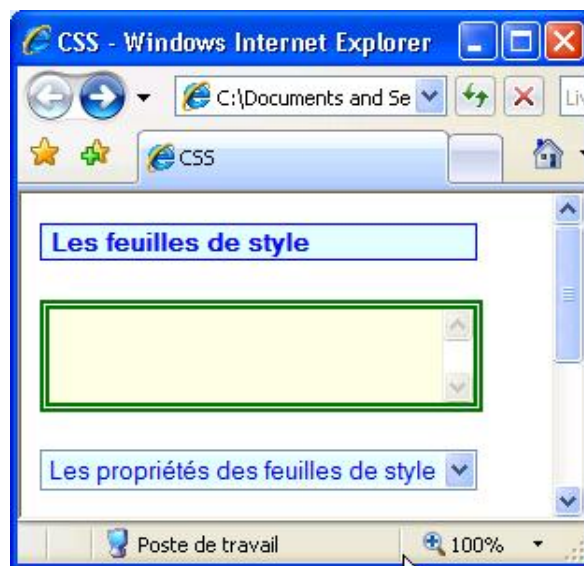
- Faites une liste de sélection avec un arrière-plan d'un bleu léger.

```

<body>
<form action="">
<select size="1" style="background-color: #elfeff; color:blue">
<option> ... etc
</select>
</form>
</body>

```

Résultat



6. Donner de la couleur à vos boutons

Exemple

```

<body>
<form action="">
<input type="submit" style="background-color:yellow; border:gray
solid 1px">
</form>
</body>

```

Résultat



7. Affecter un arrière-plan de couleur à du texte

Exemple

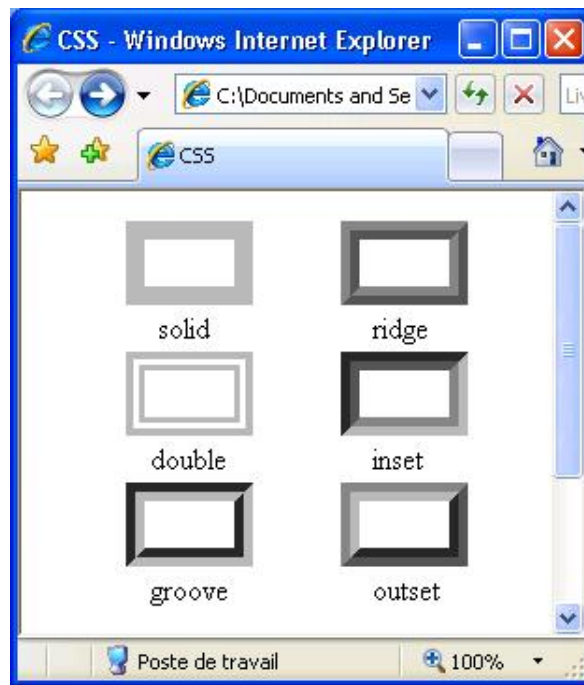
```
<body>  
<p><span style="background-color:#FFFF00">Les feuilles de style CSS  
</span></p>  
</body>
```

Résultat



8. Créer de nouvelles bordures

Les feuilles de style de bordure, comme évoqué au point Les feuilles de style de ce chapitre, peuvent prendre différentes valeurs.



border-style-solid Fait apparaître un trait en ligne continu.

border-style-double Fait apparaître un trait en ligne double.

border-style-groove Fait apparaître un trait 3D "gravé".

border-style-ridge Fait apparaître un trait 3D en relief.

border-style-inset Fait apparaître un cadre type bouton appuyé.

border-style-outset Fait apparaître cadre type bouton non appuyé.

Exemple

```
<body>
<div style="width:150px;
           height:80px;
           border:10px;
           border-style:outset;
           border-color:aqua">

</div>
</body>
```

Résultat



Bords arrondis

Si vous êtes lassé des formes rectangulaires et que vous ne souhaitez pas faire appel à des astuces graphiques, Firefox possède une propriété de style, propre, qui permet de définir le radius de l'angle du coin du rectangle de

l'élément boîte.

Cette propriété n'est pas reprise dans Internet Explorer 6 et 7. Cet exemple ne vaut donc que pour Firefox 2 et 3. Internet Explorer 6 et 7 affichera les bords de façon rectangulaire.

Exemple

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<h3 style="width: 220px;
        border: solid 5px black;
        text-align: center;
        padding: 5px;
        -moz-border-radius: 20%;">
Boite avec bords arrondis</h3>
</body>
</html>
```



9. Changer la couleur de la barre de défilement du navigateur

Internet Explorer depuis sa version 5.5, permet par des feuilles de style propres à Microsoft de modifier ou d'harmoniser la barre de défilement du navigateur. Ce genre de raffinement peut aussi être appliqué à la barre de défilement d'une zone de texte <textarea>.

À l'intérieur des balises <head> ... </head> :

Exemple

```
<style type="text/css">
body{
scrollbar-face-color:couleur;
scrollbar-arrow-color:couleur;
scrollbar-3dlight-color:couleur;
scrollbar-shadow-color:couleur;
scrollbar-highlight-color:couleur;
scrollbar-track-color:couleur;
}
</style>
```

Pour vous guider :

scrollbar-face-color:couleur

Détermine la couleur de la barre de défilement et des petites boîtes supérieures et inférieures avec les flèches. À la limite, cette propriété suffit.

scrollbar-arrow-color:couleur;

Détermine la couleur des flèches des petites boîtes supérieures et inférieures.

```
scrollbar-3dlight-color:couleur;
```

Détermine la couleur du bord, supérieur et gauche, du bouton de défilement (l'ascenseur) de la barre de défilement et des boîtes avec les flèches.

```
scrollbar-shadow-color:couleur;
```

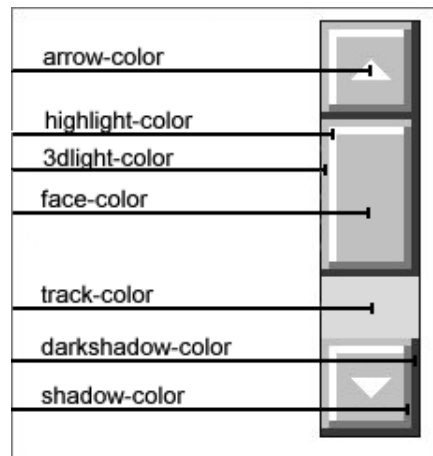
Détermine la couleur du bord inférieur et droit du bouton de défilement (l'ascenseur) de la barre de défilement et des boîtes avec les flèches.

```
scrollbar-highlight-color:couleur;
```

Détermine la couleur (légère) du fond de la barre de défilement.

```
scrollbar-track-color:couleur;
```

Détermine la couleur (dure) du fond de la barre de défilement.



Résultat possible



➤ Ces styles sont des styles propriétaires de Microsoft. Ils sont compatibles avec Internet Explorer, seulement à partir de la version 5.5. Firefox ne les reprend pas.

10. Ajouter une image qui change au survol de la souris

Il est indispensable que les deux images soient de la **même** dimension.

Exemple

```
<body>
```

```
<div onMouseOver="document.img1.src='image d'arrivée.gif'"
onMouseOut="document.img1.src='image de départ.gif'">

</div>
</body>
```

Résultat



Précautions préalables

Paradoxalement, alors que le Html n'était pas conçu à l'origine pour le multimédia, le fait de pouvoir ajouter des sons et des vidéos aux pages Web a largement contribué au succès d'Internet auprès d'une partie des internautes.

"Rien n'est plus aléatoire que l'ajout d'éléments multimédia dans une page Html" (tiré de l'aide de Dreamweaver).

En effet, sur **votre** ordinateur, avec **vos** applications, avec **votre** configuration de fichiers, les sons et vidéos programmés seront toujours activés.

Mais qu'en est-il de l'ordinateur de votre visiteur ? A-t-il Windows Media Player ou Real Audio configuré par défaut ? A-t-il WinAmp ou un autre lecteur de MP3 ? Il existe toute une série de paramètres que vous ne maîtrisez pas et vos sons ou vidéos seront parfois activés sans problèmes mais parfois aussi votre visiteur n'entendra ou ne verra rien !

De plus, il existe de nombreux formats de fichiers pour le son (wav, au, aif, mid, mp3, ra, rm...) et pour les vidéos (avi, mov, mpeg, rm...).

Pour terminer, si lors de l'étude des images (cf. Chapitre Les images et arrière-plans), nous avons attiré votre attention sur leur taille en octets, avec les sons et surtout les vidéos, le problème se pose à nouveau mais cette fois multiplié par 50 ou par 100. Les fichiers son et vidéo sont souvent d'une taille imposante et nécessitent donc un temps de chargement important. Cette notion est importante, surtout que lors de vos essais en interne le temps de réponse est immédiat !

Insérer un fond sonore

Microsoft Internet Explorer propose une balise spéciale pour faire jouer un son à l'ouverture de la page.

La balise s'écrit :

```
<bgsound src="fichier_son">
```

On peut faire jouer ce fichier son plusieurs fois par l'attribut `loop="x"` où `x` est le nombre de fois que le fichier son devra être joué. Avec `loop="infinite"`, le fichier sera exécuté en continu.

Commentaires

- Le fichier son peut avoir le format wav, au ou mid.
- Avec cette balise, il n'y a pas de problèmes de configuration à condition d'avoir Internet Explorer.
- Cette balise, qui ne fait pas partie du Html 4.0, n'est pas reconnue par Firefox.

Insérer un fichier son

Prenons pour exemple, un fichier mp3.

1. Par la balise de liens <a> ...

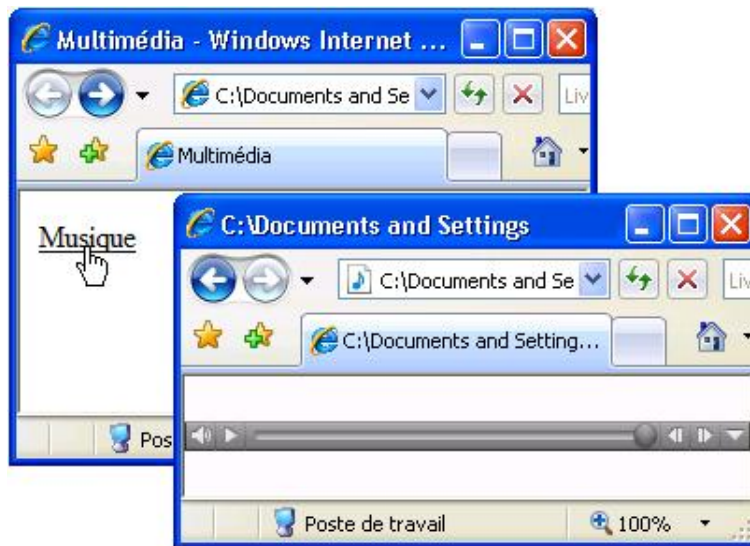
C'est la solution la plus simple mais aussi la plus hasardeuse.

```
<a href="fichier_son">Musique</a>
```



Cette balise ouvre une fenêtre **externe** pour lire le son.

Résultat



Commentaires

- Cette balise illustre bien le problème du multimédia. Partant du principe que le navigateur ne traite un fichier que si une application est associée par défaut à ce type de fichier, on peut avoir la chance que l'utilisateur final de votre page ait Windows Media Player comme application associée par défaut au fichier. Dans ce cas, tout va bien, Windows Media Player est ouvert et joue le fichier son. Par contre, si c'est l'application WinAmp qui est configurée par défaut pour ce type de fichier, le navigateur ouvrira alors WinAmp.
- Parfois encore, en cas de doute ou en cas de plusieurs applications aptes à traiter le même type de fichiers, le navigateur vous proposera la fenêtre d'invite du téléchargement. Vous devrez alors choisir d'ouvrir le fichier et parfois de déterminer le chemin vers l'application adéquate.

Pas si simple donc ... Mais la simple balise <a> est peut-être encore le moyen le plus efficace car il permettra de jouer le son sur la plupart des configurations du visiteur. C'est en tout cas la méthode la plus facile à mettre en œuvre par le webmestre débutant.

2. Par la balise <embed> ... </embed>

La balise <embed> affiche directement la console sur la page, soit en **interne** (sans passer par une fenêtre externe).

```
<embed type="audio/mp3" src="fichier_mp3"></embed>
```

Les attributs possibles sont :

align="left" ou "right" ou "top" ou "bottom"

Pour aligner la console par rapport au texte.

```
width="x"
```

Pour la largeur de la fenêtre de la console.

```
height="x"
```

Pour la hauteur de la fenêtre de la console.

```
autostart="true" ou "false"
```

Pour jouer le son directement "true" (par défaut) ou au lancement du son par l'utilisateur "false".

```
loop=true
```

Pour faire jouer le son en boucle.

```
hidden="true"
```

Pour masquer la console.

```
autoload="true"
```

Pour le téléchargement automatique dès l'affichage de la page.

```
pluginspace="URL"
```

Pour spécifier l'adresse du plug-in (facultatif). Par exemple, plugin-space="http://microsoft.com/windows/mediaplayer/en/download/".



Les formats wav, au et mp3 sont reconnus par Windows Media Player.

Il faut noter que si la balise `<embed>` est bien acceptée par les navigateurs actuels, elle n'est pas conforme à la norme Html 4.0.

3. Par la balise `<object> ... </object>`

La balise `<object> ... </object>` permet d'insérer dans la page, un objet externe au fichier Html. Cet objet est généralement de type multimédia (image, fichier audio, fichier vidéo, animation Flash, applet Java...).

Bien que la balise `<object>` soit tout à fait valide en Html 4.0, elle est la source de certaines incompatibilités. La balise `<embed>` était mieux reconnue par les différents navigateurs que ne l'est actuellement la balise `<object>`.

```
<object type="audio/mp3" data="fichier_mp3" width="280" height="45"
classid="clsid:22d6f312-b0f6-11d0-94ab-0080c74c7e95">
<param name="src" value="fichier_mp3">
<param name="autoplay" value="false">
<param name="loop" value="false">
<param name="controller" value="true">
</object>
```

L'attribut `classid` pointe vers un contrôle ActiveX de Windows. Celui-ci varie selon le lecteur multimédia :

- Pour Flash : `classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"`.
- Pour Media Player : `classid="clsid:22d6f312-b0f6-11d0-94ab-0080c74c7e95"`.
- Pour QuickTime : `classid="clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B"`.

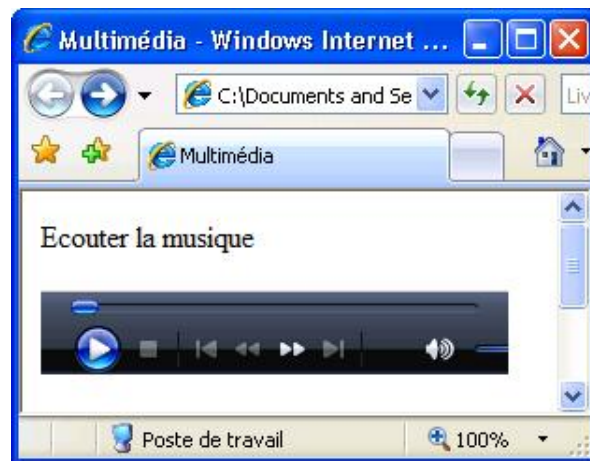
- Pour Real Player : classid="clsid:CFCDA03-8BE4-11cf-B84B-0020AFBCCFA".

Les balises <param> fournissent des paramètres pour le lecteur :

- <param name="src" value="fichier_mp3"> définit l'adresse du fichier.
- <param name="autoplay" value="false"> détermine si le fichier doit être joué directement (value="true") ou au lancement par l'utilisateur (value="false").
- <param name="loop" value="false"> fait jouer le fichier une fois (value="false") ou en boucle (value="true").
- <param name="controller" value="true"> affiche les boutons de contrôle (value="true") ou cache ceux-ci (value="false").

Ce code ne fonctionne que sous Internet Explorer 6 et 7.

Résultat



4. Avec les balises <object> et <embed>

Pour des raisons de compatibilité, il n'est pas rare que les balises <object> et <embed> soient utilisées conjointement.

Un code compatible Internet Explorer et Firefox devient :

```
<object type="audio/mp3" data="fichier_mp3" width="280" height="45"
classid="clsid:22d6f312-b0f6-11d0-94ab-0080c74c7e95">
<param name="src" value="fichier_mp3">
<param name="autoplay" value="false">
<param name="loop" value="false">
<param name="controller" value="true">
<embed width="280" height="45" type="audio/mp3" src="fichier_mp3">
</embed>
</object>
```

Insérer un fichier vidéo

La problématique est identique pour les fichiers vidéo.

Prenons comme exemple un fichier vidéo mpeg.

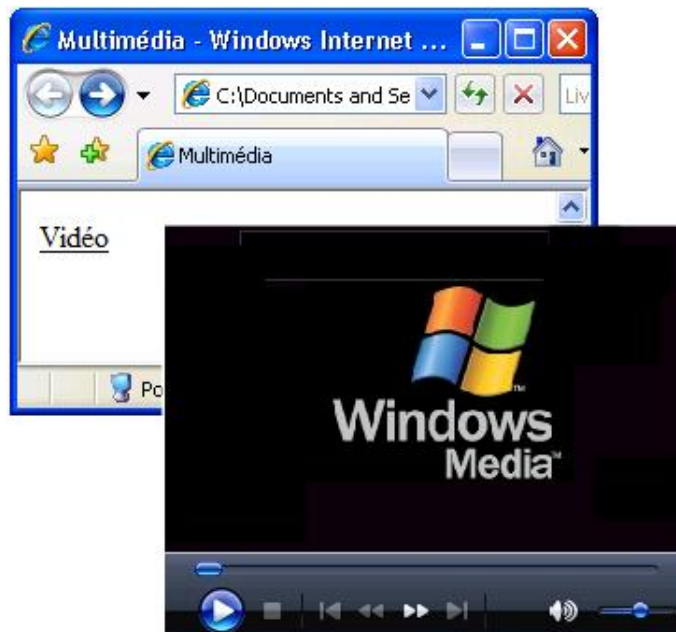
1. Par la balise de liens <a>

```
<a href="fichier_video">Film</a>
```



Cette balise ouvre une fenêtre **externe** pour lire la vidéo.

Résultat



2. Par la balise <embed> ... </embed>

La balise <embed> affiche directement la console sur la page sans passer par une fenêtre externe.

```
<embed type="video/mpeg" src="fichier_mpg"></embed>
```

Les attributs possibles sont :

align="left" OU "right" OU "top" OU "bottom"

Pour aligner la console par rapport au texte.

width="x"

Pour la largeur de la fenêtre de la console.

height="x"

Pour la hauteur de la fenêtre de la console.

autostart="true" OU "false"

Pour jouer le son directement "true" (par défaut) ou au lancement du son par l'utilisateur "false".

```
loop=true
```

Pour faire jouer le son en boucle.

```
hidden="true"
```

Pour masquer la console.

```
autoload="rue"
```

Pour le téléchargement automatique dès l'affichage de la page.

```
pluginspace="URL"
```

Pour spécifier l'adresse du plugin (facultatif). Par exemple, plugin-space="http://microsoft.com/windows/mediaplayer/en/download/".

3. Par la balise <object> ... </object>

La balise <object> ... </object> permettant d'insérer dans la page un objet externe, elle peut également être utilisée pour les fichiers vidéo.

```
<object type="video/mpeg" data="fichier_mpg" width="353" height="288"
classid="clsid:22d6f312-b0f6-11d0-94ab-0080c74c7e95">
<param name="Filename" value="fichier_mpg">
<param name="ShowControls" value="1">
<param name="ShowStatusBar" value="0">
<param name="ShowDisplay" value="0">
</object>
```

Les remarques concernant l'attribut classid et les balises <param> sont identiques à celle pour les fichiers audio.

Ce code ne fonctionne que sous Internet Explorer 6 et 7.

Exemple



4. Avec la balise <object> et <embed>

Pour des raisons de compatibilité, il n'est pas rare que les balises <object> et <embed> soient utilisées conjointement.

Un code compatible Internet Explorer et Firefox devient :

```
<object type="video/mpeg" data="fichier_mpg" width="353" height="288"
classid="clsid:22d6f312-b0f6-11d0-94ab-0080c74c7e95">
<param name="Filename" value="fichier_mpg">
<param name="ShowControls" value="1">
<param name="ShowStatusBar" value="0">
<param name="ShowDisplay" value="0">
<embed type="video/mpeg" src="fichier_mpg"></embed>
</object>
```

Insérer une animation Flash

L'application Macromedia Flash permet d'élaborer des animations complexes. Il est très facile d'en inclure dans les pages Html. Là aussi, attention au temps de chargement ! Ces applications nécessitent un plug-in "Flash Player", présent dans les versions récentes des navigateurs.

```
<object data="fichier_swf" type="application/x-shockwave-flash" width="150"
height="120">
<param name="movie" value="fichier_swf">
<param name="quality" value="high">
</object>
```

La balise `<param name="quality" value="high">` définit la qualité de l'animation (high ou low).

L'attribut `type="application/x-shockwave-flash"` détermine le type MIME des applications Flash.

Les attributs `width="x"` et `height="y"` fixent la dimension de la fenêtre d'affichage de l'animation.

Présentation

1. Le Html est mort !

Après les versions Html 1, Html 2.0, Html 3.2 et Html 4.0, le W3C a annoncé début 2000 sa décision d'arrêter les évolutions du langage Html.

Poussé par la guerre technologique que se sont livrés Netscape et Microsoft, le code Html a évolué de façon un peu anarchique et surtout désordonnée. De plus, les navigateurs, afin d'assurer la plus grande lisibilité possible des pages Web, se sont montrés de plus en plus permissifs quant au code source. Il était temps de revenir à un peu plus de rigueur.

Est-ce la faute aux éditeurs Html ? Est-ce celle des navigateurs ? Force est de constater qu'en termes de balises Html, on peut maintenant écrire n'importe quoi et n'importe comment, et que de toute façon, la page s'affiche plus ou moins convenablement dans l'un ou l'autre navigateur.

Les exemples ne manquent pas :

- La balise de fin de paragraphe `</p>` est devenue facultative.
- La stricte imbrication des balises appartient au passé.
- Dans certaines situations, un tableau peut très bien se terminer sans balise de fin de tableau.
- L'espace insécable peut se noter ` ` sous Explorer alors que la syntaxe correcte est ` `;
- L'absence de la balise de fin `</html>` n'aura pas d'incidence sur l'affichage de la page Web.
- Du simple texte enregistré sous le format .htm sera affiché sous Internet Explorer !

Pour terminer, le Html a été conçu pour les ordinateurs de type PC. Il n'est pas assez flexible (mais qui aurait pu le deviner à l'époque ?) pour prendre en compte les multiples alternatives actuelles pour accéder au Web comme les téléphones, consoles de jeux, pochettes ou mobiles.

Il devenait nécessaire d'inclure la publication Web classique dans un nouveau contexte, plus large et plus flexible mais aussi plus rigoureux, afin de pouvoir à l'avenir, le faire évoluer plus aisément en fonction des différents besoins.



Début 2008, le W3C est revenu sur sa décision de ne plus faire évoluer le Html. Une version Html 5 a été mise en chantier et un premier brouillon (*Working Draft*) est paru le 22 janvier 2008.

2. Vive le XHTML !

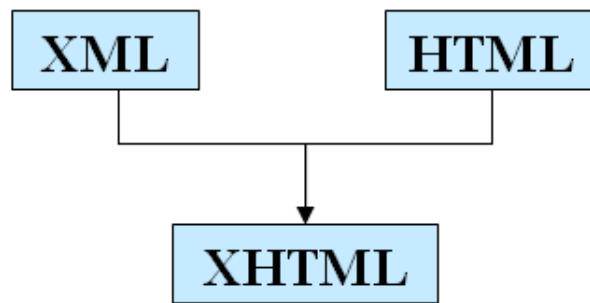
Le W3C, consortium qui réglemente la publication sur le Web, avait cependant créé entre-temps un générateur de langages (un langage pour écrire des langages ou "métalangage"), le XML, conçu à partir d'une feuille blanche pour répondre de façon évolutive aux développements futurs des multiples projets de publication électronique.

Le 26 janvier 2000 est né le XHTML 1.0 qui est présenté comme la reformulation des balises du HTML 4.0 selon les nouvelles règles de présentation et de syntaxe du XML.



Le XHTML reprend donc simplement toutes les balises et prescriptions du Html 4.0 mais avec la syntaxe assez particulière et surtout plus stricte du XML.

Cependant, dans la pratique, le XHTML se révèle être plus qu'une reformulation des balises HTML car il s'inscrit dans un contexte tout à fait novateur qu'est le XML. Nous y reviendrons au point B.



3. Constat

Faut-il abandonner les balises Html ? Faut-il dès aujourd'hui "écrire" en XHTML ?

Si votre objectif est la création d'un site amateur, le Html fait encore très bien l'affaire et ce pour de nombreuses années car les navigateurs actuels et futurs devront garder une forme de compatibilité envers les anciennes spécifications du Html.

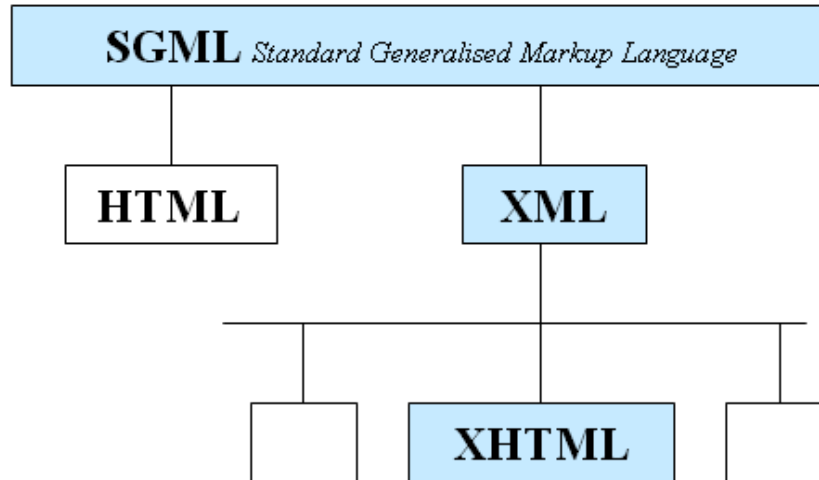
Si par contre, vous envisagez un site professionnel, il devient urgent d'envisager une migration vers le XHTML. Les arguments avancés sont :

- Un code plus rigoureux et donc plus lisible ainsi que plus facile à retravailler.
- Une meilleure compatibilité des pages quel que soit le navigateur de l'utilisateur final.
- La possibilité d'inclure et de mettre en œuvre dans les pages XHTML, des applications en XML ou de ses langages dérivés.

Le XHTML et le XML

Dans la nuit des temps de l'informatique (soit bien avant le Web), on a défini le SGML pour *Standard Generalized Markup Language* qui est un langage normalisé pour la génération de langages de balises. Cette norme internationale (ISO8879) décrit la structure et le contenu de différents types de documents électroniques. Le SGML est un langage très professionnel qui a la particularité d'être très concis et très abstrait. En conséquence, il n'est que très difficilement utilisable.

Sa descendance est pourtant assez nombreuse et l'une d'elle, qui est un langage de balises utilisé pour la publication sur le Web, vous est maintenant familière : le HTML (*HyperText Markup Language*).



Dans les années 1998-1999, le W3C a donné le jour au langage du futur sur le Web c'est-à-dire le XML (*eXtensible Markup Language*). Le XML est un métalangage, soit un ensemble de règles et de prescriptions pour permettre d'inventer de nouveaux langages (d'où le "eXtensible") avec ses propres balises. Cela signifie que XML n'est pas un langage figé comme peut l'être le Html mais au contraire un langage évolutif.

Le XHTML est un des nombreux "enfants", présents et à venir, du XML.

Le XML, tout comme son cousin le HTML, est issu du SGML qui reste le langage de référence pour la gestion électronique des documents. Cependant, il s'en rapproche davantage dans le sens où le XML est une forme accessible du SGML ; ce dernier étant décidément trop complexe pour s'afficher directement sur le Web.

Le texte simple

Après cette introduction, débutons notre étude par l'encodage du texte.

- Reprenez votre document Html minimal.

Le texte d'une page Web s'encode tout simplement dans la partie appelée le corps du document, soit entre les balises `<body>` et `</body>`.

- Dans le respect des traditions informatiques, lancez le classique "Bonjour".

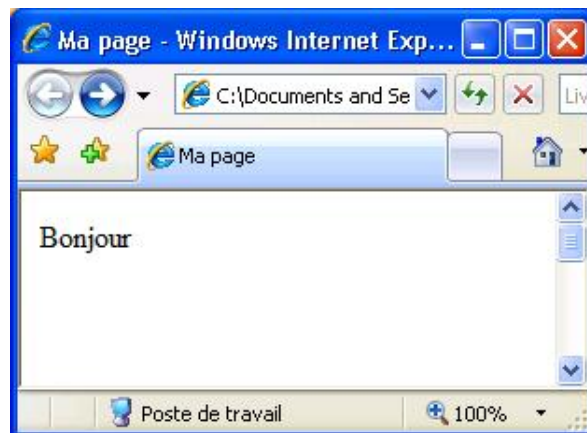
Dans un simple éditeur de texte comme le Bloc-notes de Windows ou SimpleText de Macintosh, encodez les lignes suivantes :

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Bonjour
</body>
</html>
```

Après avoir enregistré le document sous une extension .htm ou .html, ouvrez la page (en local) dans le navigateur.

Résultat



Commentaires

- Pour la suite de votre étude, il est conseillé de ne pas fermer vos applications (soit votre éditeur de texte et votre navigateur) mais de simplement les réduire.
- À chaque modification de votre document Html dans le Bloc-notes, après l'avoir enregistré bien entendu, il suffit de rappeler le navigateur. Pour voir votre fichier modifié, n'oubliez pas de cliquer sur le bouton **Actualiser** sous Internet Explorer ou **Actualiser la page courante** sous Firefox.

Les nouvelles règles de syntaxe

Le XML impose à tous ses descendants, dont le XHTML, une série de règles communes car elles appartiennent toutes à la syntaxe propre du XML.

1. Le XML est un langage strict dont il faut impérativement respecter la syntaxe. Votre document XHTML devra être "impeccable" (*well-formed*) pour éviter tout problème d'affichage, voire pas d'affichage du tout.
2. Le XML est très rigoureux quant à la structure du document. La structure de base d'un document XHTML est :

Exemple

```
<html>
<head> ... </head>
<body> ... </body>
</html>
```

Commentaire

Le document doit impérativement comporter les balises `<html>`, `<head>` et `<body>`. Les balises de titre (`<title> ... </title>`) sont obligatoires dans l'en-tête.

3. Le XML, étant sensible aux majuscules et aux minuscules (case sensitive), toutes les balises du XHTML sont souvent codées en minuscules. C'est une des raisons pour laquelle nous avons déjà noté dans cet ouvrage toutes les balises du Html en minuscules.

Exemple

XHTML	HTML
<code><p>Bonjour</p></code>	<code><P>Bonjour</p></code>
<code><h1>Titre</h1></code>	<code><H1>Titre</h1></code>

4. À toute balise d'ouverture doit correspondre une balise de fermeture. Désormais, plus de balises `<p>` ou `` orphelines...

Exemple

XHTML	HTML
<code><p>Bonjour</p></code>	<code><p>Bonjour</code>
<code></code> <code>1</code> <code>2</code> <code></code>	<code></code> <code>1</code> <code>2</code> <code></code>

5. Les balises uniques doivent également comporter une barre oblique / de fin.

L'utilisation d'une seconde balise (de fermeture cette fois) n'est pas en théorie erronée, soit par exemple `
</br>`, mais les navigateurs des anciennes générations pourraient ne pas l'interpréter correctement. C'est pourquoi `
` (avec un espace avant le /) est conseillé. Il en est de même pour les balises meta comme par exemple :

```
<meta name="http-equiv" content="pragma:no-cache" />.
```

Exemple

XHTML	HTML
<code>
</code>	<code>
</code>
<code></code>	<code></code>

6. Les balises doivent être correctement imbriquées.

Exemple

XHTML	HTML
<code><i>contenu</i></code>	<code><i>contenu</i></code>

7. La valeur des attributs doit toujours être codée entre des guillemets (et ce même pour des valeurs numériques).

Exemple

XHTML	HTML
<code><table width="100%"></code>	<code><table width=100%></code>

8. Les notations compactes ou raccourcies dans les attributs sont à éviter.

Exemple

XHTML	HTML
<code><input checked="checked"></code>	<code><input checked></code>
<code><option selected="selected"></code>	<code><option selected></code>
<code><frame noresize="noresize"></code>	<code><frame noresize></code>

9. Les balises images doivent obligatoirement comporter l'attribut alt pour le texte alternatif.

Exemple

XHTML	HTML
<code></code>	<code></code>



Il existe encore d'autres règles mais elles sortent du cadre de cet aperçu du XHTML.

Le DOCTYPE ou DTD

Une des particularités des langages de la famille du XML est de pouvoir définir ses propres balises par l'intermédiaire d'un DTD (*Document Type Declaration*). La référence à un DTD est indispensable car il contient la liste et la définition des balises utilisées dans le document. Le DTD sera toujours la seconde ligne d'un document XHTML.

Il y a pour l'instant trois types de document XHTML 1.0 :

- Strict.
- Transitional.
- Frameset.

Le W3C recommande bien entendu d'utiliser les DTDs installés sur leur propre serveur.

1. XHTML 1.0 Strict

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Comme son nom l'indique, il impose un codage strict. Il est à utiliser lorsque l'on souhaite coder proprement les balises sans effet de présentation. On recommande ce DTD lorsque l'on emploie des feuilles de style (CSS).

Le DTD Strict exclut les balises et les attributs qui ne sont plus reconnus (deprecated) depuis le Html 4.0, comme `<center>`, `<u>`, `<s>` and `<strike>`.

2. XHTML 1.0 Transitional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Il est à utiliser lorsque vous souhaitez tirer profit des apports de présentation du Html et pour les navigateurs qui ne supportent pas les feuilles de style.



Le DTD Transitional est plus souple car il inclut les balises et les attributs "deprecated".

3. XHTML 1.0 Frameset

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

Ce DTD est prévu lorsque vous souhaitez utiliser des cadres pour diviser la fenêtre du navigateur en plusieurs parties de fenêtre. En fait, le DTD Frameset inclut le DTD Transitional et les balises de cadres.



Certains pourraient s'étonner que la balise de DOCTYPE ne comporte pas de barre oblique de fin. Mais la balise DOCTYPE est du XML et n'est pas à proprement parler du XHTML ; il est donc logique qu'il n'en suive pas la syntaxe. C'est également la seule ligne du document qui comporte des majuscules et des minuscules.

Le document XHTML type

Voici un document XHTML type dûment expliqué.

```
<?xml version="1.0" encoding="UTF-8"?>
```

Cette ligne rappelle que le XHTML est issu du XML et en reprend la syntaxe.

L'attribut encoding indique au navigateur le jeu de caractères à utiliser. Par défaut, le XHTML utilise Unicode's UTF-8 ou UTF-16, mais il peut arriver qu'un autre jeu de caractères soit utilisé.

Cette ligne est facultative et il est recommandé de l'ignorer car Internet Explorer ne la prend pas bien en compte.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Le fameux DTD est repris pour indiquer le type de XHTML utilisé (Strict, Transitional ou Frameset). Attention, les majuscules et les minuscules sont à respecter scrupuleusement.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
```

Le document XHTML commence avec la balise <html> qui comporte différents attributs.

L'attribut xmlns="..." renvoie aux "namespaces" (d'où le ns) ou "espaces de nom" utilisés par le XML pour le référentiel des noms qui sera utilisé par le compilateur XML.

Le W3C recommande aussi d'identifier la langue utilisée pour le document avec la mention xml:lang="xx". L'attribut lang="xx" n'est là que pour assurer la compatibilité avec les anciens navigateurs.

```
<head>
<title>Page XHTML</title>
<meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
</head>
```

Les balises (obligatoires) d'en-tête <head> doivent nécessairement contenir un titre <title> ... </title>.

La balise méta <meta http-equiv="Content-type" content="text/html; charset=UTF-8" /> est facultative lorsque le jeu de caractères a été défini par <?xml version="1.0" encoding="UTF-8"?>. Elle est cependant conseillée pour des raisons de compatibilité car la balise XML risque de ne pas être interprétée par certains navigateurs. Par contre, la balise <meta> le sera de toute façon par tous.

Les autres balises <meta> du Html 4.0 sont admises si elles se terminent par une barre oblique de fin.

```
<body>
... corps du document ...
</body>
```

Dans le body, vous encodez les balises comme en Html 4.0, en respectant les règles de syntaxe du XHTML comme détaillées précédemment.

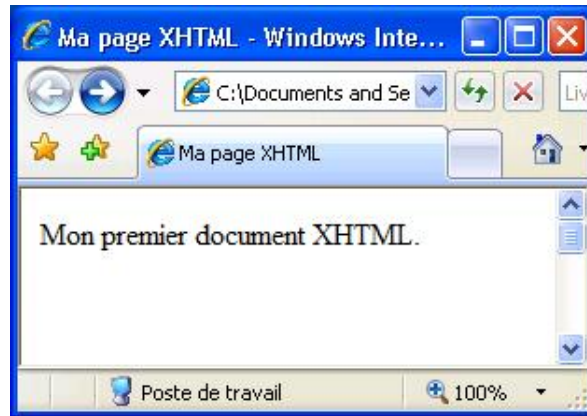
```
</html>
```

Fin du document XHTML. Cette balise de fin est indispensable.

Exemple du document type complet

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
<head>
<title>Ma page XHTML</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
</head>
<body>
Mon premier document XHTML.
</body>
</html>
```

Résultat



Commentaire

- En XHTML, on garde l'extension de fichier .htm classique.

Comparaison d'un fichier HTML et XHTML

HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Ma page XHTML</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body>
<p>Le XHTML est un langage strict<br>
qui reprend la syntaxe du XML</p>
<p align="center">

</p>
</body>
</html>
```

XHTML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
<head>
<title>Ma page XHTML</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
</head>
<body>
<p>Le XHTML est un langage strict<br />
qui reprend la syntaxe du XML</p>
<p align="center">

</p>
</body>
</html>
```

La validation du document

Étant donné la rigueur du code réclamée par le XHTML, surtout avec le doctype strict, il est prudent voire indispensable de valider les documents XHTML.

Vous pouvez vous reporter aux pages consacrées aux validateurs du chapitre L'en-tête d'un document HTML - Le DOCTYPE et la validation.

Une étude plus complète et plus détaillée du XHTML vous est proposée dans l'ouvrage XHTML et CSS dans la collection Ressources Informatiques aux Éditions ENI.

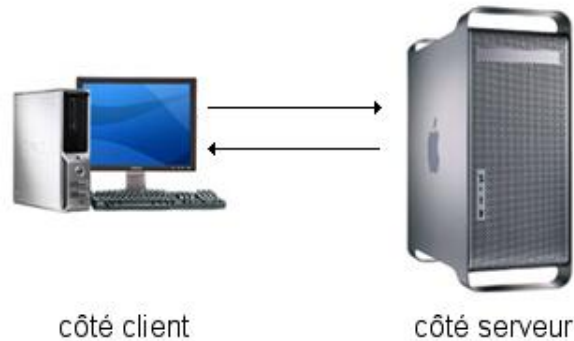
Côté client/côté serveur

Il importe pour la suite de rappeler ou de préciser le mode de fonctionnement du Web.

L'utilisateur, à partir de son navigateur (le client), effectue une requête sur l'adresse de la page Web qu'il souhaite consulter. L'ordinateur du fournisseur d'accès à Internet (le serveur) traite la demande et lui envoie le document Html correspondant.

Dans la plupart des cas, il s'agit de pages entièrement écrites en Html. Mais les concepteurs peuvent aussi adjoindre au Html, des instructions dans des langages particuliers qui seront traitées par le navigateur lui-même. Ces instructions, présentes dans le code Html et traitées directement par le navigateur, sont des applications simples, aux fonctions limitées. Ce sont des applications comme le JavaScript ou le DHTML.

Ce chapitre est donc consacré à ces langages qui viennent s'ajouter au code Html et qui sont gérées directement par le navigateur. Chacun de ses langages nécessiterait une étude approfondie, notre objectif sera simplement d'en découvrir les richesses et potentialités pour une utilisation ponctuelle.



Des instructions dans de véritables langages de programmation peuvent être adjointes au code Html pour réaliser des opérations bien plus complexes, comme par exemple une boutique en ligne ou une consultation de bases de données ; mais il est évident que ces applications ne peuvent pas être prises en charge directement par les navigateurs des utilisateurs. Ces instructions doivent alors être préalablement traitées par le serveur avant d'être envoyées au client. Ces applications complexes aux fonctions étendues mises en place à l'aide de langages tels que le PHP, l'ASP, le CGI, le Perl,... dépassent en complexité le cadre de cet ouvrage et de l'état présent de vos connaissances en matière de publication sur le Web.

Le JavaScript

1. Présentation

Le Html, prévu pour mettre en ligne des documents d'essence textuelle, est par définition statique. Très vite (soit dans les années 95/96), sous l'impulsion de Netscape, le besoin s'est fait sentir d'ajouter des éléments dynamiques et de l'interactivité dans les pages Web. Après quelques balbutiements (LiveScript) est apparu le langage JavaScript qui a connu depuis de multiples évolutions. Il a en outre été repris par Microsoft sous le nom de JScript.

Le JavaScript est devenu maintenant un standard dans l'univers de la publication sur le Web sous le nom de ECMAScript et est reconnu par la quasi-totalité des navigateurs actuels.

Voici quelques-unes de ses caractéristiques :

- Le JavaScript s'intègre dans les pages et s'ajoute au code source Html. Il fonctionne en quelque sorte comme une extension du langage Html.
- Le JavaScript est défini comme un langage de scripts, tout comme le Html. Mais JavaScript est bel et bien un langage de programmation, simpliste pour les programmeurs confirmés mais assez complexe pour ceux dont la programmation est un domaine inconnu. JavaScript est inspiré du langage C dont il reprend le concept objet avec toutes les notions de variables, opérateurs, fonctions, tests conditionnels, etc.
- Les codes du JavaScript, qui s'insèrent dans les documents Html, sont gérés et exécutés par le navigateur lui-même sans devoir faire appel aux ressources du serveur (côté client). Les instructions sont donc traitées en direct et surtout sans retard (car il ne faut pas faire appel aux ressources du serveur) par le navigateur ; ce qui présente l'avantage de ne pas ralentir l'affichage des pages. Par contre, pour des raisons de sécurité, les possibilités du JavaScript ont été volontairement limitées.
- Comme les codes du JavaScript s'intègrent dans les pages Web, ils sont visibles par une simple consultation du code source de la page ; ce qui fait le désespoir de certains concepteurs débutants mais qui constitue un handicap dans le cadre professionnel. D'autres langages plus élaborés comme le PHP ou l'ASP, gérés directement sur le serveur (côté serveur), permettent des applications qui préserveront la confidentialité du code.
- JavaScript, étant un langage orienté objet, permet d'accéder directement aux éléments de la page et du navigateur. Il permet ainsi par exemple de reconnaître la résolution d'écran du visiteur, de détecter le type et la version du navigateur, de lire et de traiter les données d'un formulaire, d'échanger des informations entre des cadres, etc.

En résumé, JavaScript :

- Apporte de l'interactivité aux pages Web.
- Est reconnu par presque tous les navigateurs.
- Est du code intégré dans le document Html.
- Est traité directement par le navigateur.
- Est un langage de programmation simple.
- Permet d'intervenir directement sur les éléments (objets) du document et du navigateur.
- Laisse le code source des scripts visible.

Le JavaScript étant interprété par le navigateur, n'importe quel navigateur compatible JavaScript peut être adopté. Ce qui est le cas de la plupart des navigateurs du marché, Internet Explorer 6 et 7 ainsi que Firefox 2 et 3, retenus dans le cadre de cet ouvrage, sont bien entendu parfaitement adaptés au JavaScript. Veillez cependant à ce que la fonction JavaScript du navigateur ne soit pas désactivée.

2. Un premier script

Le JavaScript comporte des boîtes de message d'alerte.

Élaborez un premier script (élémentaire).

Exemple

```
<html>
<head>
<title>Mon premier script</title>
<script type="text/JavaScript">
<!--
function message() {
alert("Bonjour !");
}
// -->
</script>
</head>
<body>
<form>
<input type="button" value="Test" onClick="message()">
</form>
</body>
</html>
```

Résultat



Commentaires

1. Les instructions JavaScript doivent être comprises entre les balises `<script> ... /script>` pour être prises en charge et traitées par le navigateur.

La balise `<script>` est complétée par l'attribut `type="text/javascript"` qui détermine qu'il s'agit d'un script JavaScript. En effet, il existe d'autres langages de script qui peuvent s'implémenter dans une page Html, tel que le VBScript de Microsoft qui est un langage issu du Visual Basic mais qui n'a pas connu la même diffusion que le JavaScript.

2. Les balises `<!-- ... //-->` sont destinées à cacher les lignes de code aux navigateurs qui ne reconnaissent pas le JavaScript et à leur empêcher d'afficher le code sous forme de texte. Il est raisonnable de penser qu'à l'heure actuelle les navigateurs qui ne reconnaissent pas le JavaScript sont plutôt rares et elles sont devenues facultatives.

3. Le code `function message()` introduit une fonction JavaScript à qui nous avons donné le nom de `message`. Les lignes de code d'une fonction commencent toujours par une accolade ouvrante `{` et se terminent par une accolade fermante `}`.

4. Une boîte de message spécifique à JavaScript (appelée par "alert") a été utilisée. Comme le contenu à afficher dans la boîte de message est du texte, celui-ci doit être mis entre des guillemets ou des apostrophes. Si votre texte comporte des apostrophes, il faudra encoder `\'` afin qu'elles ne soient pas interprétées comme du code.

5. À la fin de chaque instruction JavaScript des points-virgules sont notifiés (ce qui n'est pas sans rappeler le C et le C++). Le JavaScript est moins strict que ces autres langages et ne signale généralement pas de message d'erreur

s'ils venaient à manquer. On peut considérer que le point-virgule est optionnel et qu'il n'est obligatoire que lorsque vous écrivez plusieurs instructions sur une même ligne. Il est toutefois vivement recommandé d'en mettre de façon systématique.

6. Il vous sera peut-être utile d'inclure des commentaires personnels dans vos lignes de code, JavaScript utilise les conventions utilisées en C et C++ soit // commentaire. Ainsi tout ce qui est écrit entre le // et la fin de la ligne sera ignoré. Il est aussi possible d'inclure des commentaires sur plusieurs lignes avec le code /* commentaire sur plusieurs lignes */

Attention à ne pas confondre les commentaires JavaScript et les commentaires Html (<!-- ... -->).

7. Dans le corps du document, on retrouve un bouton (cf. Chapitre Les formulaires).

8. La balise <input> est complétée par ce qu'on appelle en JavaScript un déclencheur d'événement ici, onClick soit au clic de la souris. Le clic de la souris (onClick) appelle la fonction message définie précédemment dans la page.

3. Scripts internes et scripts externes

Le code JavaScript peut s'inclure directement dans le document Html (scripts internes) ou faire l'objet d'un fichier distinct (scripts externes).

a. Scripts internes

Il n'y a pas de directives strictes pour inclure un script dans un document. Il est toutefois préférable de respecter les règles ci-après.

Le navigateur traite la page Html de haut en bas (y compris vos ajouts en JavaScript). Par conséquent, toute instruction ne pourra être exécutée que si le navigateur possède à ce moment précis tous les éléments nécessaires à son exécution. Ces éléments doivent donc être déclarés avant ou au plus tard lors de l'instruction.

Pour s'assurer que le programme de script est chargé dans la page et prêt à fonctionner à toute intervention de votre visiteur, déclarez systématiquement (lorsque cela sera possible) un maximum d'éléments dans les balises d'en-tête soit entre <head> et </head> et avant la balise <body>.

Rien n'interdit de mettre plusieurs scripts dans une même page Html.

Il faut noter que l'usage de la balise <script> n'est pas toujours obligatoire. Ce sera le cas des événements JavaScript (par exemple onClick) où il suffit d'insérer le code à l'intérieur de la commande Html comme un attribut. L'événement fera appel à la fonction JavaScript lorsque la commande Html sera activée. JavaScript fonctionne alors comme une extension du langage Html.

Dans l'exemple précédent le code deviendrait :

```
<html>
<head>
<title>Mon premier script</title>
</head>
<body>
<form action="">
<input type="button" value="Test" onClick="alert('Bonjour !')">
</form>
</body>
</html>
```

b. Scripts externes

Il est possible d'utiliser des fichiers externes pour les programmes JavaScript. On peut ainsi stocker les scripts dans des fichiers distincts (avec l'extension .js) et les appeler à partir d'un fichier Html.

Cette façon de procéder offre l'avantage de pouvoir appeler un même script par différentes pages Html et de cacher momentanément le code aux internautes débutants.

Pour réaliser le fichier de script externe, il suffit d'encoder dans un éditeur de texte :

```
function message() {
alert("Bonjour !");
}
```

et de l'enregistrer sous l'extension .js soit ici message.js.

Dans le fichier de départ, le script externe sera appelé par la balise `<script src="script_externe"></script>`.

```
<html>
<head>
<title>Mon premier script</title>
<script src="message.js"></script>
</head>
<body>
<form action="">
<input type="button" value=" Test " onClick="message()">
</form>
</body>
</html>
```

4. Ouvrir une nouvelle fenêtre

Pour continuer notre approche pragmatique de JavaScript, il est peut-être intéressant d'étudier l'objet `window`, qui permet d'ouvrir ou de fermer une mini-fenêtre : il s'agit de ces fameuses fenêtres pop-up.

L'objet `window` a deux méthodes :

`open()` pour ouvrir une fenêtre.

`close()` pour fermer la fenêtre en cours.

La syntaxe est :

```
[window.]open( "URL" , "nom" , "caractéristiques" );
```

Commentaires

- `window` est facultatif car c'est l'objet par défaut en JavaScript.
- L'URL est l'adresse de la page que l'on désire afficher dans la nouvelle fenêtre.
- `Caractéristiques` est une liste de certaines ou de toutes les caractéristiques de fenêtre que l'on note à la suite, séparées par des virgules et sans espaces ni passage à la ligne.

Ces caractéristiques sont :

`toolbar=yes` ou `no`

Affichage de la barre d'outils.

`location=yes` ou `non`

Affichage de champ d'adresse.

`status=yes` ou `no`

Affichage de la barre d'état.

`menubar=yes` ou `no`

Affichage de la barre de menus.

`scrollbars=yes` ou `no`

Affichage des barres de défilement.

`resizable=yes` ou `no`

Dimension de la fenêtre modifiable.

width=x en pixels

Largeur de la fenêtre en pixels.

height=y en pixels

Hauteur de la fenêtre en pixels.



On peut aussi utiliser 1 ou 0 au lieu de yes ou no.



L'usage de ces fenêtres est assez sympathique pour afficher des informations complémentaires sans surcharger la page (ou fenêtre) de départ. Mais surtout, prévoyez toujours un moyen pour l'utilisateur de fermer cette fenêtre pop-up. À utiliser avec modération cependant !

- Ouvrez une nouvelle fenêtre qui contiendra le fichier popup.htm à partir d'un lien dans la page de départ.

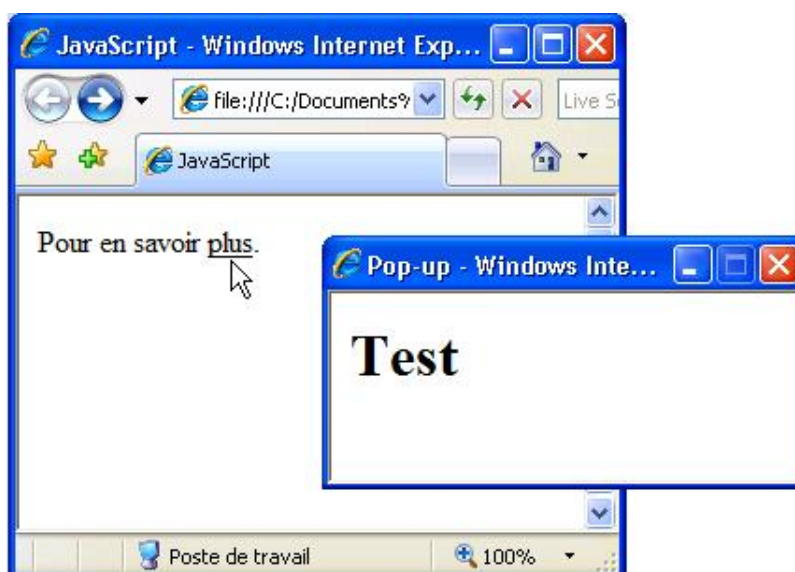
Le fichier popup.htm :

```
<html>
<head>
<title>Pop-up</title>
</head>
<body>
<h1>Test</h1>
</body>
</html>
```

Le fichier de départ :

```
<html>
<head>
<title>JavaScript</title>
</head>
<body>
Pour en savoir <a href="#" onClick="open('popup.htm', '', 'width=120,
height=100,toolbar=no,location=no,status=no,menubar=no,
scrollbars=no,resizable=no') ">plus</a>
</body>
</html>
```

Résultat



Commentaires

- Attention, il est impératif que tout le contenu de `onClick` soit écrit sans espaces ni passage à la ligne (ce qui n'était pas possible en version papier).

- Ajoutez maintenant un bouton de fermeture à la fenêtre pop-up.

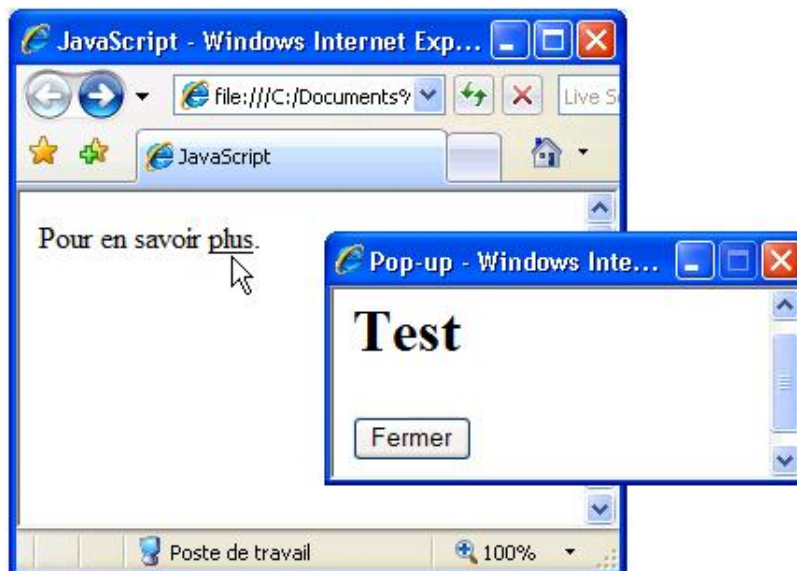
Le fichier `popup.htm` devient alors `popup1.htm`.

```
<html>
<head>
<title>Pop-up</title>
</head>
<body>
<h1>Test</h1>
<form>
<input type="button" value="Fermer" onClick="self.close()">
</form>
</body>
</html>
```

Profitez-en pour modifier quelques caractéristiques de la fenêtre pop-up. Le lien qui lance la fenêtre pourrait être :

```
<html>
<head>
<title>JavaScript</title>
</head>
<body>
Pour en savoir <a href="#" onClick="open('popup1.htm', '',
'width=120,height=100,toolbar=no,location=no,status=no,menubar=no,
scrollbars=yes,
resizable=no')">plus</a>
</body>
</html>
```

Résultat



Commentaire

- Attention, toujours sans espaces, ni passage à la ligne !
- Refermez automatiquement la fenêtre pop-up après quelques secondes... (JavaScript offre la possibilité d'incorporer un compteur pour ne déclencher une action qu'après un petit laps de temps).

La syntaxe du compteur est :

```
nom_du_compteur = setTimeout("fonction_appelée", temps en milliseconde)
```

Incorporez un compteur dans le fichier popup.htm :

```
compt=setTimeout("self.close();",4000).
```

Ce compteur dont le nom est `compt` déclenchera la fonction de fermeture `self.close()` après 4 secondes.

Il faut encore déclencher la mise en route du compteur. Pour ce faire, profitez du chargement de la page (l'événement `onLoad`). Ajoutez à la balise `<body>` du fichier popup.htm, la ligne suivante :

```
<body onLoad='compt=setTimeout("self.close();",4000) '>
```

Le fichier complet de popup2.htm est donc :

```
<html>
<head>
<title>Pop-up</title>
</head>
<body onLoad='compt=setTimeout("self.close();",4000) '>
<h1>Test</h1>
</body>
</html>
```

5. Traiter les données d'un formulaire

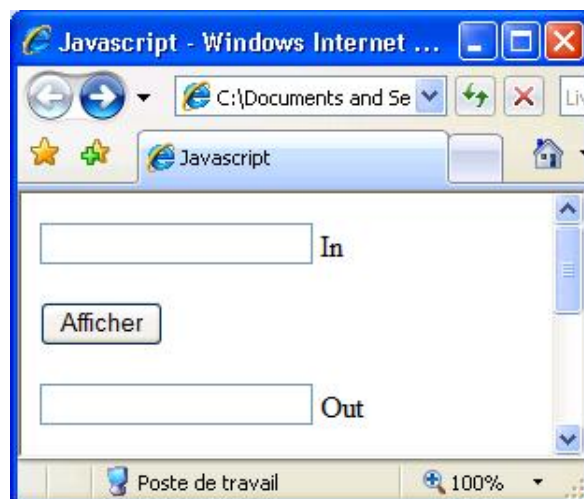
Construisez d'abord un formulaire avec deux lignes de texte et un bouton comme vu au chapitre consacré aux formulaires.

L'attribut `name` à la balise `<form>` et `<input type="text">` aux deux balises de ligne de texte ont bien été indiqués.

Exemple

```
<html>
<head>
<title>Javascript</title>
</head>
<body>
<form name="form" action="">
<input type="text" name="input" value=""> In
<p>
<input type="button" value="Afficher">
</p>
<input type="text" name="output" value=""> Out
</form>
</body>
</html>
```

Résultat



Le but de l'exercice est, après avoir appuyé sur le bouton **Afficher**, de recopier les données de la ligne de texte **In** dans la zone de texte **Out**. Cette opération sera réalisée par la fonction `afficher()` que l'on placera dans l'en-tête :

```
<script type="text/JavaScript">
function afficher(form) {
var data=document.form.input.value;
document.form.output.value=data
}
</script>
```

Commentaires

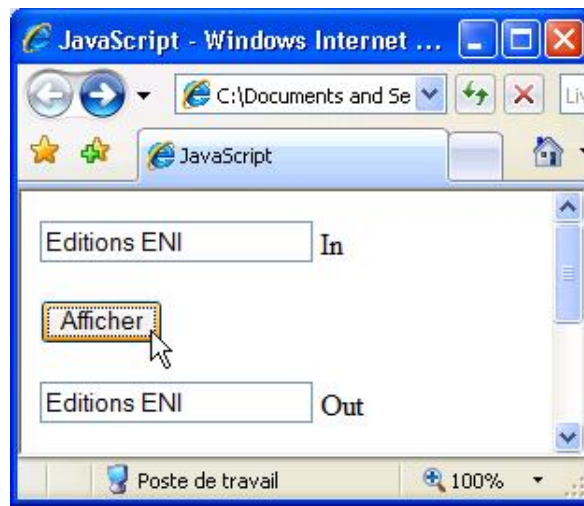
- Pour procéder, la fonction `afficher()` a besoin des données du formulaire `form`. Ces données sont transmises comme argument à la fonction entre les parenthèses de celle-ci.
- La variable (`var data`) définie contiendra les données du formulaire "In".
- Ces données se trouvent dans `document.form.input.value`. Voici une belle illustration du concept objets de JavaScript et de leur hiérarchisation. On part toujours de l'objet le plus extérieur vers l'objet le plus intérieur. Pour indiquer le chemin à JavaScript, il doit partir du document (`document`) ; dans ce document, il existe un formulaire (`form`), ce formulaire contient une ligne de texte (`input`) et la valeur (`value`).
- On donne alors à la zone de texte **Out** la valeur de la variable `data`. Le chemin vers **Out** s'élabore de la même façon. Dans le document (`document`), il y a un formulaire (`form`). Dans ce formulaire se trouve la ligne de texte de sortie (`output`) qui contient un attribut (`value`) pour recevoir la valeur.

Il faut encore ajouter le déclencheur d'événement `onClick="afficher (form)"` au bouton **Afficher**. Le contenu du formulaire (`form`) est transmis comme argument à la fonction `afficher()`.

Le fichier complet avec le script est donc :

```
<html>
<head>
<title>JavaScript</title>
<script type="text/JavaScript">
function afficher(form) {
var data=document.form.input.value;
document.form.output.value=data
}
</script>
</head>
<body>
<form name="form" action="">
<input type="text" name="input" value=""> In
<p>
<input type="button" value="Afficher" onClick="afficher(form)">
</p>
<input type="text" name="output" value=""> Out
</form>
</body>
</html>
```

Résultat



6. Quelques scripts utiles

Avec ces connaissances élémentaires, il vous sera possible d'utiliser utilement les quelques scripts suivants.

a. Un bouton de confirmation ou d'annulation

Une autre boîte de message `confirm()` de JavaScript affiche deux boutons : **OK** et **Annuler**.

Elle renvoie en outre une valeur (booléenne) qui est égale à "true" en cas de clic sur **OK** et à "false" en cas de clic sur **Annuler**, ce qui est pratique pour un test conditionnel.

Exemple

```
<script>
function ok(){
if(confirm("Voulez-vous continuer ?")){
location.href = "suite.htm";
}
else{
null
}
}
}
</script>
```

Résultat



Commentaires

- Lorsque le visiteur clique sur le bouton **OK**, il est transporté à la page suite.htm. S'il clique sur **Annuler**, rien ne se passe (null).
- Il ne faut pas oublier de mettre le déclencheur `onClick="ok()"` dans une balise de lien.

b. Détection de la résolution d'écran de votre visiteur

Un petit script permettra de détecter la résolution d'écran de votre visiteur pour le rediriger vers une page conçue

pour elle.

On ajoute dans les balises <head> ... </head> :

Exemple

```
<script type="text/JavaScript">
if (screen.width==800) {
if(confirm("Redirection vers la version 800x600."))
window.location="800x600.htm";
}
else
if (screen.width==1280) {
if(confirm("Redirection vers la version 1280x1024."))
window.location="1280x1024.htm";
}
}</script>
```

Résultat



Commentaire

- Nous supposons que le site a été optimisé pour une résolution 1024x768. Si la largeur est de 800, le visiteur est redirigé avec une boîte de confirmation (voir exemple précédent) vers une page prévue pour la résolution 800x600. Par contre, si la largeur est de 1280, il est redirigé vers une page en 1280x1024.

c. Détection de la version du navigateur

```
<body>
<script type="text/JavaScript">
document.write("Le code name de votre navigateur est " +navigator.
appName );
document.write("Le nom ou la marque du navigateur est "
+navigator.appName ); document.write("Les informations sur la version
sont "+navigator.appVersion); document.write("Le navigateur a comme
user-agent name "+navigator.userAgent);
</script>
</body>
```

Pour Explorer 7.0, on obtient :

- Le code name de votre navigateur est Mozilla.
- Le nom ou la marque du navigateur est Microsoft Internet Explorer.
- Les informations sur la version sont 4.0 (compatible; MSIE 7.0; Windows NT 5.1; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) ; .NET CLR 1.1.4322; .NET CLR 2.0.50727).
- Le navigateur a comme user-agent name Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) ; .NET CLR 1.1.4322; .NET CLR 2.0.50727).

Pour Explorer 6.0, on obtient :

- Le code name de votre navigateur est Mozilla.

- Le nom ou la marque du navigateur est Microsoft Internet Explorer.
- Les informations sur la version sont 4.0 (compatible; MSIE 6.0; Windows NT 5.1; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) ; .NET CLR 1.1.4322; .NET CLR 2.0.50727).
- Le navigateur a comme user-agent Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) ; .NET CLR 1.1.4322; .NET CLR 2.0.50727).

Pour Firefox 2, on obtient :

- Le code name de votre navigateur est Mozilla.
- Le nom ou la marque du navigateur est Netscape.
- Les informations sur la version sont 5.0 (Windows; fr-FR).
- Le navigateur a comme user-agent name Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.8.1.14) Gecko/20080404 Firefox/2.0.0.14.

Pour Firefox 3, on obtient :

- Le code name de votre navigateur est Mozilla.
- Le nom ou la marque du navigateur est Netscape.
- Les informations sur la version sont 5.0 (Windows; fr).
- Le navigateur a comme user-agent Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9) Gecko/2008051206 Firefox/3.0.

d. Détection du navigateur par le concept objet

Le concept objet s'est développé au fil des évolutions des navigateurs. Ainsi, il suffit de tester l'existence de certains objets pour connaître le type et/ou la version du navigateur.

Ainsi on utilise :

```
document.all
```

Le navigateur utilisé est Internet Explorer 4 ou plus.

```
document.getElementById
```

Le navigateur utilisé est Internet Explorer 5 ou plus.

```
window.createPopup
```

Le navigateur utilisé est Internet Explorer 5.5 ou plus.

```
document.compatMode && document.all
```

Le navigateur utilisé est Internet Explorer 6 ou plus.

```
typeof document.body.style.maxHeight != "undefined"
```

Le navigateur utilisé est Internet Explorer 7 ou plus.

```
window.getComputedStyle
```

Le navigateur utilisé est Firefox 1 ou plus.

```
window.Iterator
```

Le navigateur utilisé est Firefox 2 ou plus.

Le script se place dans l'en-tête, donc entre <head> ... </head>.

```
<script type="text/javascript">
```

```

if (document.compatMode && document.all) {
window.location="explorer6.htm";
}
if (typeof document.body.style.maxHeight != "undefined") {
window.location="explorer7.htm";
}
if (window.Iterator) {
window.location="firefox.htm";
}
}
</script>

```

e. Afficher la date et l'heure

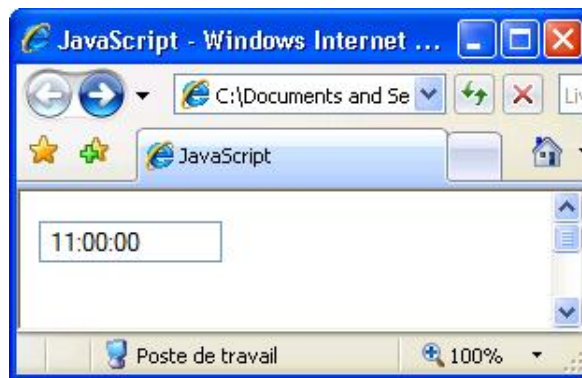
Exemple

```

<html>
<head>
<script type="text/JavaScript">
function getDt(){
dt=new Date();
hrs=dt.getHours();
min=dt.getMinutes();
sec=dt.getSeconds();
tm=" "+((hrs<10)?"0":"")+hrs+":";
tm+=((min<10)?"0":"")+min+":";
tm+=((sec<10)?"0":"")+sec+" ";
document.horloge.display.value=tm;
setTimeout("getDt()",1000);
}
</script>
</head>
<body onLoad="getDt()">
<form name="horloge">
<input type="text" name="display" size="12" value ="">
</form>
</body>
</html>

```

Résultat



Commentaire

- Le script utilise des fonctions JavaScript de l'objet Date comme `getHours()`, `getMinutes()` ou `getSeconds()`. Un compteur `setTime-out` est inséré pour ajouter l'effet de trotteuse des secondes (non visible en version papier). Et le tout fonctionne dans une ligne de texte.

f. Un texte défilant

Exemple

```

<html>
<head>

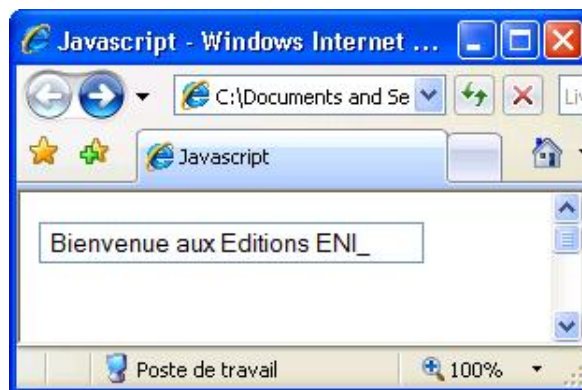
```

```

<script type="text/JavaScript">
var m = " Bienvenue aux Editions ENI "
var a = ""
i=1
function scrollit() {
i++
if(i==m.length) {
null}
else{
if(i > m.length) {
i = 1
}
a = m.substring(0,i)
document.form.zone.value = a+"_"
setTimeout("scrollit()",100)
}
}
}
</script>
</head>
<body onLoad="scrollit()">
<form name=form>
<input type="text" size="30" name="zone" value="">
</form>
</body>
</html>

```

Résultat



g. Une image qui change au survol de la souris

Toujours spectaculaire cet effet où l'image change au survol de la souris. Il en existe différentes versions.

Voici la nôtre. Les deux images doivent avoir rigoureusement la même dimension.

Exemple

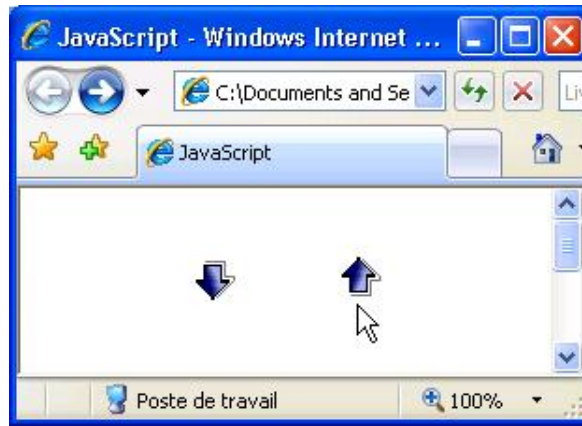
```

<html>
<head>
<script type="text/JavaScript">
function Up() {
document.images["change"].src="up.gif";
}
function Down() {
document.images["change"].src="down.gif";
}
</script>
</head>
<body>
<a href="#" onMouseOver="Up();" onMouseOut="Down();">

</a>
</body>
</html>

```


Résultat



Commentaires

- On détermine deux fonctions : l'une `up()` concerne l'image à afficher lors du survol de la souris (`onmouseover`) et l'autre `down()` est l'image "au repos" à laquelle on revient lorsque la souris quitte la zone de l'image (`onmouseout`).
- Les attributs `width` et `height` de la balise `` sont indispensables en JavaScript.

h. Donner le focus à un formulaire

Il est sympathique, lorsque la page comporte un formulaire, de donner d'entrée le focus (le curseur qui clignote) à un élément de formulaire.

Le focus étant un élément de JavaScript, il suffit dans un petit script de lui indiquer le chemin vers l'élément du formulaire soit `document.form.nom.focus()`.

Exemple

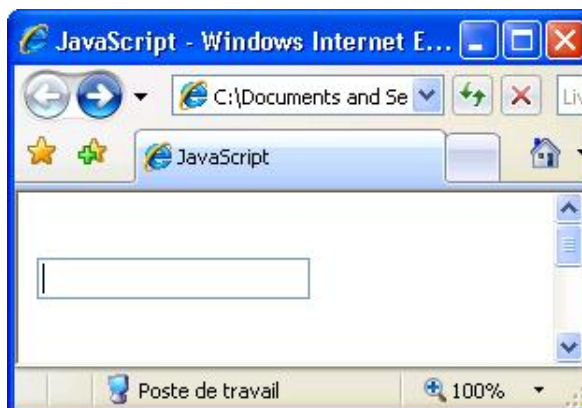
Soit le formulaire :

```
<form name="form" action="">
<input type="text" name="nom">
</form>
```

on ajoute juste après le formulaire :

```
<script type="text/JavaScript">
document.form.nom.focus();
</script>
```

Résultat



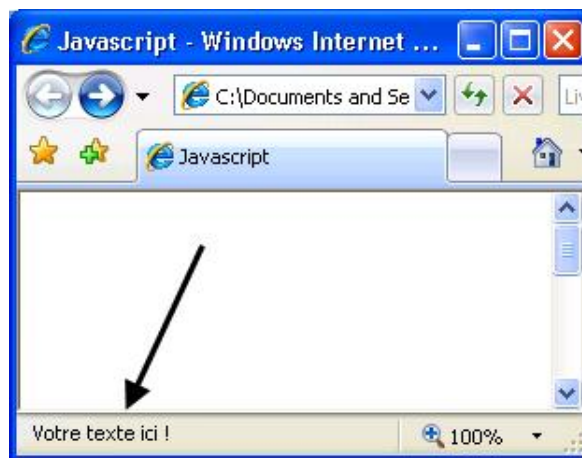
Le curseur qui clignote (focus) est bien dans la ligne de texte.

i. Afficher du texte dans la barre de statut

Exemple

```
<html>
<head>
<script type="text/JavaScript">
var message="Votre texte ici !";
var visible=0;
function Flash() {
if (visible == 0) {
window.status=message;
visible=1;
}
else {
window.status="";
visible=0;
}
setTimeout('Flash()', 400);
}
</script>
</head>
<body onLoad="Flash()">
</body>
</html>
```

Résultat



Commentaire

- Le texte apparaît bien au bas de la fenêtre dans la barre d'état. Et en plus, même si vous ne le voyez pas sur papier, il clignote !

j. Sortir des cadres

Lorsque votre site est appelé à partir d'un autre site comportant des cadres, vos pages risquent de se retrouver un peu à l'étroit dans une des fenêtres du site appelant et de ne pas disposer de tout l'espace que vous lui avez destiné.

Ce problème peut être résolu par ce petit JavaScript à insérer dans toutes les pages de base de votre site. Votre site apparaîtra alors toujours dans une fenêtre normale.

Recopiez simplement le script suivant dans la partie <head>:

```
<script type="text/JavaScript">
<!--
if (parent.frames.length!=0)
parent.location.href=location.href
// -->
</script>
```

7. Continuer votre étude

Une étude plus complète et plus détaillée du langage JavaScript vous est proposée dans l'ouvrage "Des CSS au DHTML" dans la collection Expert IT des Éditions ENI.

Le DHTML

1. Présentation

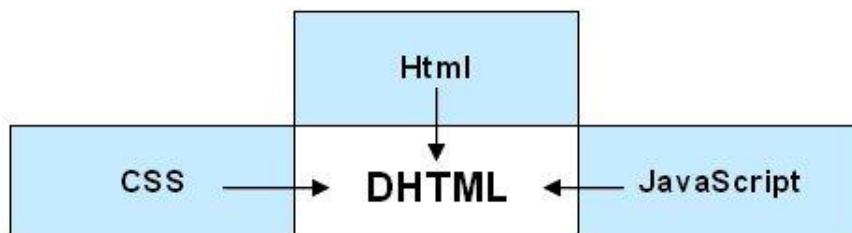
Le DHTML (*Dynamic HTML*) n'est pas un langage de balises de plus, ni un langage de script ou de programmation. C'est simplement une forme d'écriture et de conception de pages Web où l'accent a été mis, dans des proportions variables cependant, sur certaines formes d'animation et d'interactivité.

Le Dhtml permet de concevoir des pages Html dont les éléments peuvent être modifiés après que celles-ci aient été chargées par le navigateur (sans faire appel aux ressources du serveur).

À son apparition (fin 1997/début 1998), on définissait le Dhtml comme une technique permettant d'apporter de façon dynamique des animations aux pages Html classiques.

En effet, les pages Html sont, par définition, statiques et surtout plus orientées vers la richesse de contenu que vers l'attrait visuel. Le Dhtml permet d'apporter quelques fantaisies graphiques pour en améliorer l'attrait.

Le Dhtml utilise trois développements de la publication sur la toile ; le Html, le JavaScript et les feuilles de style CSS.



Depuis quelques années, le panorama des outils de publication a bien évolué avec l'émergence des feuilles de style. Et le Dhtml a retrouvé une nouvelle jeunesse.

Les publications récentes n'hésitent pas à définir le Dhtml comme le moyen de modifier dynamiquement les déclarations de style des éléments du Html au moyen du JavaScript.

Le Dhtml est du JavaScript appliqué aux feuilles de style CSS.

Cette définition, techniquement plus précise, semble rallier l'approbation de tous. Surtout que le Dhtml a délaissé au fil du temps ses animations parfois un peu futiles pour se concentrer sur l'amélioration de la convivialité et de l'ergonomie des sites Web.

2. En pratique

Ces scripts Dhtml sont généralement très complexes et nécessitent des connaissances pointues en JavaScript et en feuilles de style CSS. Une solution consiste à reprendre (en respectant le copyright éventuel) des scripts disponibles sur le Web. Une recherche avec les mots-clés "script Dhtml", vous mènera sur des sites qui vous proposeront des scripts qu'il suffira d'incorporer dans le code source de votre page par copier/coller.

Il est important :

- De vérifier les conditions de compatibilité du script avec les navigateurs et leurs versions. Les sites bien conçus ne manqueront pas de vous le signaler.
- De respecter, à la lettre, les instructions de fonctionnement et de personnalisation.

3. Quelques exemples

a. Modifier la taille de la police de caractères

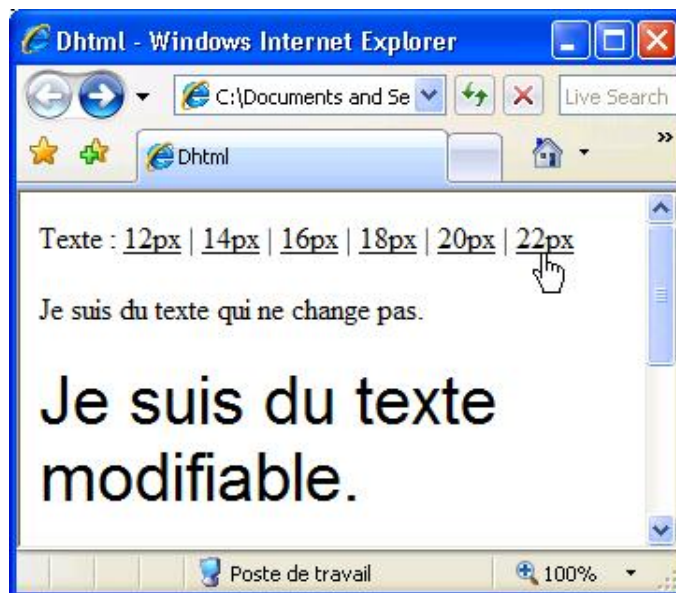
Ce script permet au visiteur malvoyant de choisir lui-même la taille du texte. Par un clic de souris, choisit la taille de caractères qui lui convient le mieux parmi plusieurs propositions.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">
```

```

<html lang="fr">
<head>
<title>Dhtml</title>
<style type="text/css">
.mod{ font-family:sans-serif;}
</style>
<script type="text/javascript">
function getElementsByClass(searchClass, node, tag) {
var classElements = new Array();
if ( node == null )
node = document;
if ( tag == null )
tag = '*';
var els = node.getElementsByTagName(tag);
var elsLen = els.length;
var pattern = new RegExp("(^|\\s)" + searchClass + "(\\s|$)");
for (i = 0, j = 0; i < elsLen; i++) {
if ( pattern.test(els[i].className) ) {
classElements[j] = els[i];
j++;
}
}
return classElements;
}
function taillePolice(classe, taille) {
body = document.getElementsByTagName('body');
cibles = getElementsByClass('mod');
for (i=0; i < cibles.length; i++) {
cibles[i].style.fontSize = taille;
}
}
</script>
</head>
<body>
<div>Texte :
<a href="" onClick="taillePolice('mod', 12); return false">12px</a> |
<a href="" onClick="taillePolice('mod', 14); return false">14px</a> |
<a href="" onClick="taillePolice('mod', 16); return false">16px</a> |
<a href="" onClick="taillePolice('mod', 18); return false">18px</a> |
<a href="" onClick="taillePolice('mod', 20); return false">20px</a> |
<a href="" onClick="taillePolice('mod', 36); return false">22px</a>
</div>
<p>Je suis du texte qui ne change pas.</p>
<p class="mod">Je suis du texte modifiable.</p>
</body>
</html>

```



b. Un menu déroulant vertical

Les menus de navigations sont un élément essentiel dans l'architecture d'un site Web. Ce script, avec un effet déroulant permet un gain de place appréciable pour les menus de navigation comportant de nouveaux items.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Dhtml</title>
<script type="text/javascript">
function montrer(id) {
var objet = document.getElementById(id);
for (var i = 1; i<=10; i++) {
if (document.getElementById('sousmenu'+i)) {
document.getElementById('sousmenu'+i).style.display='none';
}
}
if (objet) {
objet.style.display='block';
}
}
</script>
<style type="text/css">
dl, dt, dd, ul, li { margin: 0px; padding: 0px;
list-style-type: none;}
#menu { position: absolute;
top: 5; left: 10;}
dl#menu { width: 140px;}
dl#menu dt { cursor: pointer;
margin: 2px 0;
height: 20px;
line-height: 20px;
text-align: center;
font :sans-serif 10pt;
border: 1px solid black;
background: #9cf;}
dl#menu dd { border: 1px solid black;}
dl#menu li { text-align: center;
background: #fff;}
dl#menu li a { color: #000;
text-decoration: none;
display: block;
border: 0 none;
height: 100%;}
dl#menu dt a { color: #000;
text-decoration: none;
display: block;
border: 0 none;
height: 100%;}
dl#menu li a:hover { background: #9cf;}
dl#menu dt a:hover { background: #9cf;}
</style>
</head>
<body onload="montrer()">
<dl id="menu">
<dt onclick="javascript:montrer('sousmenu1');">Chapitre 1</dt>
<dd id="sousmenu1">
<ul>
<li><a href="#">Point 1</a></li>
<li><a href="#">Point 2</a></li>
<li><a href="#">Point 3</a></li>
</ul>
</dd>
<dt onclick="javascript:montrer('sousmenu2');">Chapitre 2</dt>
<dd id="sousmenu2">
```

```

<ul>
<li><a href="#">Point 1</a></li>
<li><a href="#">Point 2</a></li>
<li><a href="#">Point 3</a></li>
</ul>
</dd>
<dt onclick="javascript:montrer('sousmenu3');">Chapitre 3</dt>
<dd id="sousmenu3">
<ul>
<li><a href="#">Point 1</a></li>
<li><a href="#">Point 2</a></li>
<li><a href="#">Point 3</a></li>
</ul>
</dd>
</dl>
</body>
</html>

```



4. Continuer votre étude

Une étude plus complète et plus détaillée du DHTML vous est proposée dans l'ouvrage **Des CSS au DHTML** dans la collection Expert IT des Éditions ENI.

Les caractères spéciaux

Caractère	Code	Entity
		
,	‚	
<i>f</i>	ƒ	&fnof ;
„	„	
...	…	&hellip ;
†	†	&dagger ;
‡	‡	&Dagger ;
^	ˆ	&circ ;
‰	‰	&permil ;
Š	Š	&Scaron ;
<	‹	< ;
Œ	Œ	&Oelig ;
		
	Ž	
		
		
`	‘	
'	’	
“	“	
”	”	
•	•	&bull ;
-	–	&ndash ;
—	—	
~	˜	&tilde ;
™	™	&trade ;
š	š	&scaron ;
>	›	> ;

œ	œ	&oelig ;
		
	ž	
ÿ	Ÿ	&Yuml ;
espace	 	
i	¡	¡
ç	¢	¢
£	£	£
¤	¤	¤
¥	¥	¥
	¦	¦
§	§	§
..	¨	¨
©	©	©
ª	ª	ª
«	«	"
¬	¬	¬
-	­	­
®	®	®
-	¯	¯
°	°	°
±	±	±
2	²	²
3	³	³
'	´	´
µ	µ	µ
¶	¶	¶
.	·	·
¸	¸	¸
1	¹	¹

º	º	º
»	»	»
¼	¼	¼
½	½	½
¾	¾	¾
¿	¿	¿
À	À	À
Á	Á	Á
Â	Â	Â
Ã	Ã	Ã
Ä	Ä	Ä
Å	Å	Å
Æ	Æ	Æ
Ç	Ç	Ç
È	È	È
É	É	É
Ê	Ê	Ê
Ë	Ë	Ë
Ì	Ì	Ì
Í	Í	Í
Î	Î	Î
Ï	Ï	Ï
Ð	Ð	Ð
Ñ	Ñ	Ñ
Ò	Ò	Ò
Ó	Ó	Ó
Ô	Ô	Ô
Õ	Õ	Õ
Ö	Ö	Ö
×	×	×

Ø	Ø	Ø
Ù	Ù	Ù
Ú	Ú	Ú
Û	Û	Û
Ü	Ü	Ü
Ý	Ý	Ý
Þ	Þ	þ ;
ß	ß	ß
à	à	à
á	á	á
â	â	â
ã	ã	ã
ä	ä	ä
å	å	å
æ	æ	æ
ç	ç	ç
è	è	è
é	é	é
ê	ê	ê
ë	ë	ë
ì	ì	ì
í	í	í
î	î	î
ï	ï	ï
ð	ð	ð
ñ	ñ	ñ
ò	ò	ò
ó	ó	ó
ô	ô	ô

õ	õ	õ
ö	ö	ö
÷	÷	÷
Ø	ø	ø
ù	ù	ù
ú	ú	ú
û	û	û
ü	ü	ü
ý	ý	ý
þ	þ	þ
ÿ	ÿ	ÿ

Conversion décimal vers hexadécimal

Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex
0	00	32	20	64	40	96	60
1	01	33	22	65	41	97	61
2	02	34	23	66	42	98	62
3	03	35	24	67	43	99	63
4	04	36	25	68	44	100	64
5	05	37	26	69	45	101	65
6	06	38	27	70	46	102	66
7	07	39	28	71	47	103	67
8	08	40	29	72	48	104	68
9	09	41	2A	73	49	105	69
10	0A	42	2B	74	4A	106	6A
11	0B	43	2C	75	4B	107	6B
12	0C	44	2D	76	4C	108	6C
13	0D	45	2E	77	4D	109	6D
14	0E	46	2F	78	4E	110	6E
15	0F	47	2G	79	4F	111	6F
16	10	48	30	80	50	112	70
17	11	49	31	81	51	113	71
18	12	50	32	82	52	114	72
19	13	51	33	83	53	115	73
20	14	52	34	84	54	116	74
21	15	53	35	85	55	117	75
22	16	54	36	86	56	118	76
23	17	55	37	87	57	119	77
24	18	56	38	88	58	120	78
25	19	57	39	89	59	121	79
26	1A	58	3A	90	60	122	7A
27	1B	59	3B	91	61	123	7B

28	1C	60	3C	92	62	124	7C
29	1D	61	3D	93	63	125	7D
30	1E	62	3E	94	64	126	7E
31	1F	63	3F	95	65	127	7F
128	80	160	A0	192	C0	224	E0
129	81	161	A1	193	C1	225	E1
130	82	162	A2	194	C2	226	E2
131	83	163	A3	195	C3	227	E3
132	84	164	A4	196	C4	228	E4
133	85	165	A5	197	C5	229	E5
134	86	166	A6	198	C6	230	E6
135	87	167	A7	199	C7	231	E7
136	88	168	A8	200	C8	232	E8
137	89	169	A9	201	C9	233	E9
138	8A	170	AA	202	CA	234	EA
139	8B	171	AB	203	CB	235	EB
140	8C	172	AC	204	CC	236	EC
141	8D	173	AD	205	CD	237	ED
142	8E	174	AE	206	CE	238	EE
143	8F	175	AF	207	CF	239	EF
144	90	176	B0	208	D0	240	F0
145	91	177	B1	209	D1	241	F1
146	92	178	B2	210	D2	242	F2
147	93	179	B3	211	D3	243	F3
148	94	180	B4	212	D4	244	F4
149	95	181	B5	213	D5	245	F5
150	96	182	B6	214	D6	246	F6
151	97	183	B7	215	D7	247	F7
152	98	184	B8	216	D8	248	F8

153	99	185	B9	217	D9	249	F9
154	9A	186	BA	218	DA	250	FA
155	9B	187	BB	219	DB	251	FB
156	9C	188	BC	220	DC	252	FC
157	9D	189	BD	221	DD	253	FD
158	9E	190	BE	222	DE	254	FE
159	9F	191	BF	223	DF	255	FF

Les titres

Quoi de plus utile que de prévoir différentes tailles de caractères pour les titres du contenu textuel (par exemple, les noms de chapitre).

Le langage Html propose jusqu'à 6 grandeurs de caractères pour les titres. Quand on sait que titre en anglais se dit "heading", l'élaboration de la balise devient évidente ; <h pour heading suivi du numéro du niveau 1, 2, 3, 4, 5, 6>. Par la suite, la notation <hx> est utilisée, où x est un niveau de 1 à 6.

Explorez ces balises dans un document Html :

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<h1>Bonjour</h1>
<h2>Bonjour</h2>
<h3>Bonjour</h3>
<h4>Bonjour</h4>
<h5>Bonjour</h5>
<h6>Bonjour</h6>
</body>
</html>
```

Résultat



Commentaires

Malgré la simplicité de cette balise, plusieurs remarques s'imposent :

- Par défaut, avec la balise <hx> le texte est mis en gras.
- Par défaut également, la balise ajoute une ligne vide entre le texte de la balise et la suite du document.
- Pour obtenir des caractères plus grands que ceux proposés par le niveau 6, il faut utiliser les feuilles de styles (cf. Chapitre Les feuilles de style) ou construire des images.
- Les balises de titre sont des éléments importants pour structurer de façon efficace le contenu textuel de la

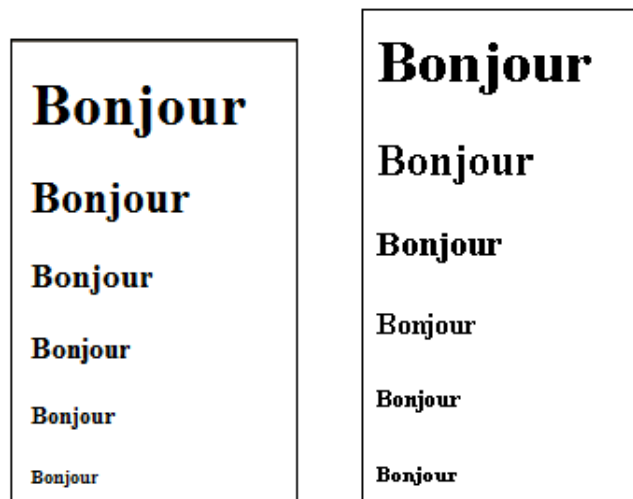
page.

- L'usage des balises de titres doit être réservé à leur fonction première, c'est-à-dire de déterminer des titres de niveaux différents. Il importe de ne pas les détourner de leur but premier pour introduire, par exemple, une mise en forme (gras) ou une taille de caractères déterminée.

Ayons la curiosité d'ouvrir cette même page dans Firefox.



Il n'y a bien entendu pas de différences essentielles pour cette balise élémentaire. Pourtant, en y regardant de plus près, l'affichage diffère entre Internet Explorer et Firefox.



Ainsi :

- La marge entre le texte et le bord supérieur de la fenêtre du navigateur est plus grande sous Internet Explorer que sous Firefox.
- La marge entre le texte et le bord gauche de la fenêtre du navigateur est légèrement plus grande sous Internet Explorer que sous Firefox.
- L'interligne est plus petit sous Internet Explorer que sous Firefox.

- Le niveau h6 est un peu moins petit et plus lisible sous Firefox que sous Internet Explorer.
- La page est plus longue sous Firefox que sous Internet Explorer.

Ces différences entre un code identique et des navigateurs différents ne manquent pas de dérouter le débutant. Elles sont monnaie courante pour le concepteur confirmé. Il ne faut jamais oublier qu'en langage Html, on n'a pas la maîtrise de l'affichage de la page. Celle-ci dépend du navigateur utilisé !

Le corps du document

1. La balise <body>

`alink="#$$$$$"`

Ajoute une couleur au lien actif.

`background="fichier_image"`

Ajoute une image d'arrière-plan.

`bgcolor="#$$$$$"` où `bgcolor="nom"`

Ajoute une couleur d'arrière-plan.

`link="#$$$$$"`

Ajoute une couleur au lien non activé.

`text="#$$$$$"`

Ajoute une couleur au texte par défaut.

`vlink="#$$$$$"`

Ajoute une couleur au lien visité.

Le texte et sa présentation

1. Le texte

`<address> ... </address>`

Signale une adresse. A pour effet de mettre en italique.

` ... `

Met le texte en gras.

`<big> ... </big>`

Augmente la taille du texte.

`<blockquote> ... </blockquote>`

Introduit un retrait pour une citation.

`
`

Force le passage à la ligne.

`<center> ... </center>`

Réalise un alignement centré. Cette balise ne fait plus partie des balises standards depuis le Html 4.0.

`<cite> ... </cite>`

Signale une citation. Le texte est affiché avec une police à pas fixe.

`<code> ... </code>`

Signale du code informatique. Le texte est affiché avec une police à pas fixe.

` ... `

Met le texte en valeur en le mettant en italique.

` ... `

` ... `

Définit la couleur du texte en valeur hexadécimale ou par nom de couleur.

` ... `

Détermine la police de caractères à utiliser.

` ... `

Indique la taille des caractères où x a une valeur de 1 à 7 ou de -3 à +3.

`<hx> ... </hx>`

Balise dite de titre où x a une valeur de 1 à 6.

`<i> ... </i>`

Met le texte en italique.

`<kbd> ... </kbd>`

Signale une saisie clavier. Le texte est affiché avec une police à pas fixe.

`<nobr> ... </nobr>`

Empêche la rupture de ligne automatique.

`<pre> ... </pre>`

Affiche le contenu en police à pas fixe tel qu'il apparaît à l'écran.

`<samp> ... </samp>`

Signale un exemple. Le texte est affiché avec une police à pas fixe.

`<small> ... </small>`

Diminue la taille du texte.

`<strike> ... </strike>`

Barre le texte en son milieu.

` ... `

Renforce le texte en le mettant en gras.

`_{...}`

Met le contenu en indice.

`^{...}`

Met le contenu en exposant.

`<tt> ... </tt>`

Affiche le texte avec une police à pas fixe.

`<u> ... </u>`

Souligne le texte.

`<var> ... </var>`

Signale une variable (le texte est affiché avec une police à pas fixe).

2. Les listes

`<dd> ... </dd>`

Introduit une description avec un retrait (élément d'une liste de définition).

`<dl> ... </dl>`

Introduit une liste de définition.

`<dt> ... </dt>`

Introduit un terme de définition (élément d'une liste de définition).

` ... `

Élément d'une liste numérotée ou d'une liste à puce.

` ... `

Introduit une liste numérotée.

` ... `

Introduit une liste à puces ou non, numérotée.

Les tableaux

1. La balise de tableau <table>

`align="left" OU "center" OU "right"`

Permet d'aligner un tableau à gauche, au centre ou à droite.

`background="fichier_image"`

Permet d'insérer une image d'arrière-plan au tableau.

`bgcolor="couleur"`

Permet d'attribuer une couleur de fond au tableau.

`border="x" où x est un nombre exprimé en pixels.`

Permet de mettre une bordure autour du tableau.

`bordercolor="couleur"`

Permet d'attribuer une couleur à la bordure du tableau.

`cellpadding="x" où x est un nombre exprimé en pixels.`

Permet de gérer l'espace entre la bordure et le contenu des cellules.

`cellspacing="x" où x est un nombre exprimé en pixels.`

Permet de gérer l'espace entre les cellules du tableau.

`height="x" OU "x%" où x est un nombre et x% est un pourcentage.`

Permet de déterminer la hauteur du tableau.

`frame="above" OU "below" OU "box" OU "hsides" OU "lhs" OU "rhs" OU "vsides" OU "void"`

Permet de choisir les côtés extérieurs du tableau auxquels affecter une bordure (cf. Chapitre Les tableaux. Présenter d'autres bordures). Ne fonctionne que sous Internet Explorer !

`rules="all" OU "cols" OU "groups" OU "none" OU "rows"`

Permet de choisir les côtés intérieurs du tableau auxquels affecter une bordure (cf. Chapitre Les tableaux. Présenter d'autres bordures). Ne fonctionne que sous Internet Explorer !

`width="x" OU "x%" où x est un nombre et x% est un pourcentage.`

Permet de déterminer la largeur du tableau.

2. La balise de ligne <tr>

`align="left" OU "center" OU "right"`

Permet d'aligner le contenu des cellules de la ligne à gauche, au centre ou à droite.

`bgcolor=couleur`

Permet d'attribuer une couleur de fond à la ligne.

`bordercolor=couleur`

Permet d'attribuer une couleur à la bordure de la ligne.

`height="x" ou "x%"` où x est un nombre et x% est un pourcentage.

Permet de déterminer la hauteur de la ligne.

`valign="bottom" ou "center" ou "top"`

Permet d'aligner le texte dans les cellules de la ligne en bas ou au centre ou en haut.

3. La balise de cellule **<td>**

`align="left" ou "center" ou "right"`

Permet d'aligner le contenu de la cellule à gauche, au centre ou à droite.

`background="fichier_image"`

Permet d'insérer une image d'arrière-plan à la cellule.

`bgcolor=couleur`

Permet d'attribuer une couleur de fond à la cellule.

`bordercolor=couleur`

Permet d'attribuer une couleur à la bordure de la cellule.

`colspan="x" où x est un nombre`

Permet de fusionner x colonnes d'un tableau.

`height="x" ou "x%"` où x est un nombre et x% est un pourcentage.

Permet de déterminer la hauteur de la cellule.

`nowrap`

Permet de désactiver la coupure de ligne automatique.

`rowspan="x" où x est un nombre`

Permet de fusionner x lignes d'un tableau.

`valign="bottom" ou "center" ou "top"`

Permet d'aligner le contenu de la cellule en bas ou au centre ou en haut.

`width="x" ou "x%"` où x est un nombre et x% est un pourcentage.

Permet de déterminer la largeur d'une cellule.

Les liens

1. La balise <a>

```
href="fichier.htm"
```

Lien vers un fichier htm situé dans le même dossier.

```
href="chemin_du_fichier"
```

Lien vers un fichier htm situé dans un autre dossier. Voir les règles de l'adressage relatif (cf. chapitre Les liens - Les liens vers une autre page).

```
href="http://www.lehtml.com/index.htm"
```

Lien vers une page située sur un autre site sur le Web.

```
href="mailto:adresse_électronique"
```

Ouvre l'application de messagerie électronique avec l'adresse électronique du destinataire.

```
target="_self" ou "_top" ou "_blank"
```

Ouvre la page cible du lien dans la même fenêtre (_self et _top) ou dans une nouvelle instance du navigateur (_blank).

```
title="Texte explicatif"
```

Ajoute une info-bulle au lien.

Les images

1. La balise

`align="top" OU "texttop" OU "absmiddle" OU "middle" OU "absbottom" OU "bottom" OU "baseline"`

Aligne l'image par rapport au texte.

`alt="Texte explicatif"`

Initialement prévu comme texte de remplacement, ajoute une info-bulle explicative à l'image. Est obligatoire depuis le Html 4.0.

`border="x"`

Spécifie l'épaisseur de la bordure. La valeur `border="0"` est souvent utilisée pour enlever le bord ajouté par défaut aux images de lien.

`height="x"`

Spécifie la hauteur de l'image (souvent accompagné de son compère `width="x"`).

`hspace="x"`

Spécifie l'espace (vide) ajouté horizontalement, soit les côtés gauche et droit, de l'image.

`lowsrc="fichier_image"`

Spécifie le nom et éventuellement l'adressage d'une image en basse résolution (et donc au temps de chargement plus rapide) qui s'affiche avant l'image déterminée par l'attribut `src`.

`scr="fichier_image"`

Spécifie le nom et éventuellement l'adressage de l'image.

`vspace="x"`

Spécifie l'espace (vide) ajouté verticalement, soit au-dessus et en dessous de l'image.

`width="x"`

Spécifie la largeur de l'image. Souvent accompagné de son compère `height="x"`.

Les cadres

1. Les attributs de la balise <frameset>

`border="x"`

Permet de changer l'épaisseur de l'ensemble des bordures des cadres. La valeur `x` est exprimée en pixels.

`bordercolor="couleur"`

Permet de déterminer la couleur de l'ensemble des bordures du jeu de cadres. Cet attribut peut également se trouver dans la commande `<frame>`. Si l'attribut se trouve dans la commande `<frame>`, il a préséance sur celui se trouvant dans la commande `<frameset>`.

`cols="liste de valeurs de largeur des colonnes"`

Indique les différentes largeurs des colonnes. Les données exprimées en % ou en pixels sont séparées par des virgules. On peut utiliser une valeur `*` pour déterminer l'espace restant.

`frameborder="1" ou "0" ("yes" ou "no")`

Permet de déterminer si les cadres ont ou n'ont pas de bordure.

`framespacing="x"`

Permet de déterminer l'espace entre les cadres. La valeur `x` est exprimée en pixels.

`rows="liste de valeurs de la hauteur des rangées"`

Indique les différentes hauteurs des rangées ou lignes. Les données exprimées en pourcentage ou en pixels sont séparées par des virgules. On peut utiliser une valeur `*` pour déterminer l'espace restant.

2. Les attributs de la balise <frame>

`border="x"`

Permet de changer l'épaisseur de la bordure du cadre. La valeur `x` est exprimée en pixels.

`bordercolor="couleur"`

Permet de spécifier la couleur de la bordure.

`frameborder="1" ou "0" ("yes" ou "no")`

Permet de déterminer si le cadre a ou n'a pas de bordure.

`framespacing="x"`

Permet d'ajuster l'espace entre les cadres. La valeur `x` doit être exprimée en pixels.

`marginheight="x"`

Spécifie la grandeur des marges de haut et de bas d'un cadre donné. La valeur `x` spécifiée doit être exprimée en pixels. Par défaut, les navigateurs déterminent une marge de 8 pixels.

`marginwidth="x"`

Spécifie la grandeur des marges de gauche et de droite d'un cadre donné. La valeur `x` spécifiée doit être exprimée en

pixels. Par défaut, les navigateurs déterminent une marge de 8 pixels.

`name="nom"`

Permet de donner un nom à un cadre. De cette façon, on peut employer ce nom comme cible (target) d'un lien provenant d'autres documents.

`noresize`

Empêche l'utilisateur de redimensionner le cadre. L'attribut `noresize` est optionnel et, par défaut tous les cadres peuvent être redimensionnés.

`scrolling="yes" OU "no" OU "auto"`

Permet d'attribuer ou non une barre de défilement à un cadre. Une valeur `"yes"` indique que la barre de défilement est toujours visible sur le cadre en question. Par contre, la valeur `"no"` fait en sorte que la barre de défilement ne soit jamais visible. La valeur `"auto"` laisse le navigateur déterminer si la barre de défilement est nécessaire. L'attribut `scrolling` est optionnel et la valeur par défaut est `"auto"`.

`src="fichier.htm"`

Cet attribut indique le fichier Html ou l'adresse du fichier Html qui sera affiché(e) dans un cadre spécifique.

3. Les attributs de la balise `<iframe>`

`align="absbottom" OU "absmiddle" OU "baseline" OU "bottom" OU "left" OU "middle" OU "right" OU "texttop" OU "top"`

Spécifie un alignement du cadre dans le document.

`border="x"`

Permet de changer l'épaisseur de la bordure du cadre. La valeur `x` est exprimée en pixels.

`bordercolor="couleur"`

Permet de spécifier la couleur de la bordure.

`frameborder="1" OU "0"`

Spécifie la présence ou non d'une bordure autour du cadre. La valeur `1` affiche une bordure (par défaut). La valeur `0` n'affiche pas de bordure.

`height="x"`

Détermine la hauteur du cadre, en pixels.

`marginheight="x"`

Détermine la hauteur de la marge de haut et de bas du cadre, en pixels.

`marginwidth="x"`

Détermine la largeur de la marge de gauche et de droite du cadre, en pixels.

`noresize`

Spécifie que l'utilisateur ne pourra pas redimensionner le cadre.

`scrolling="yes" OU "no" OU "auto"`

Permet d'attribuer ou non une barre de défilement à un cadre. Une valeur yes indique que la barre de défilement est toujours visible sur le cadre en question. Par contre, la valeur no fait en sorte que la barre de défilement ne soit jamais visible. La valeur auto laisse le navigateur déterminer si la barre de défilement est nécessaire. L'attribut scrolling est optionnel et la valeur par défaut est auto.

```
src="fichier.htm"
```

Cet attribut indique le fichier Html ou l'adresse du fichier Html qui sera affiché(e) dans un cadre spécifique.

```
width="x"
```

Spécifie la largeur du cadre en pixels.

```
hspace="x"
```

Détermine l'espace en pixels à gauche et à droite du cadre.

```
vspace="x"
```

Détermine l'espace en pixels en haut et en bas du cadre.

Les formulaires

1. Les différentes balises intervenant dans les formulaires

```
<fieldset> ... </fieldset>
```

Permet de regrouper des éléments d'un formulaire.

```
<form> ... </form>
```

Déclare un formulaire.

```
<input type="text">
```

Déclare une zone de texte d'une seule ligne.

```
<input type="radio">
```

Déclare des boutons à choix unique.

```
<input type="checkbox">
```

Déclare des boutons à choix multiples.

```
<input type="submit">
```

Déclare un bouton d'envoi.

```
<input type="reset">
```

Déclare un bouton d'annulation.

```
<input type="hidden">
```

Déclare un élément de formulaire caché.

```
<input type="file">
```

Déclare un élément de téléchargement de fichiers.

```
<input type="password">
```

Déclare une zone de texte où l'encodage des données se fait sous la forme de mots de passe.

```
<label> ... </label>
```

Associe un énoncé avec un élément de formulaire (étiquette).

```
<legend> ... </legend>
```

Permet d'ajouter une légende à des éléments de formulaires regroupés par la balise `<fieldset>`.

```
<option> ... </option>
```

Introduit un élément d'une liste de sélection déclarée par la balise `<select>`.

```
<select> ... </select>
```

Déclare une liste de sélection, souvent déroulante.

```
<textarea> ... </textarea>
```

Déclare une zone de texte à plusieurs lignes.

Les feuilles de style

1. Les feuilles de style de police (font)

font-family

- définit un nom de police ou une famille de police.
- nom de police précise (Arial, Times, Helvetica...) ou famille (serif, sans-serif, cursive, fantasy, monospace).
- `h3 {font-family:Arial}`

font-size

- définit la taille de la police.
- xx-small ou x-small ou small ou médium ou large ou x-large ou xx-large. ou larger ou smaller.ou taille précise en points (pt), inches (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `p {font-size:72pt}`

font-style

- définit le style de l'écriture.
- normal ou italic (pour italique) ou oblique.
- `h3 {font-style:italic}`

font-variant

- écrit de façon normale ou en petites majuscules.
- normal ou small-caps.
- `p {font-variant:small-caps}`

font-weight

- définit l'épaisseur de la police ou du caractère.
- normal ou bold (gras) ou bolder ou lighter ou valeur numérique soit (100|200|300|400|500|600|700|800|900).
- `p {font-weight:bold}`

font

- raccourci pour les différentes propriétés de police.
- exemple : ``

2. Les feuilles de style de texte (text)

text-align

- définit l'alignement du texte à gauche, au centre, à droite ou justifié.
- left ou center ou right ou justify.
- h1 {text-align:center}

text-indent

- définit un retrait dans la première ligne d'un bloc de texte.
- spécifié en inches (in) ou en centimètres (cm) ou en pixels (px).
- p {text-indent:1cm}

text-decoration

- définit une "décoration" du texte, soit souligné, barré, surligné, clignotant ou sans.
- underline ou line-through ou overline ou blink ou none.
- a:visited {text-decoration:blink}

text-transform

- définit la casse du texte (majuscule, minuscule).
- uppercase (met les caractères en majuscules) ou lowercase (met les caractères en minuscules) ou capitalize (met le premier caractère en majuscule).
- p {text-transform:uppercase}

line-height

- définit l'interligne soit l'espace entre les lignes du texte.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- p {line-height:10pt}

letter-spacing

- définit l'espace entre les lettres.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- p {letter-spacing:2pt}

color

- définit la couleur du texte.
- par exemple en hexadécimal.
- h3 {color:#000080}

width

- détermine la longueur d'un élément.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `h1 {width:200px}`

height

- détermine la hauteur d'un élément.
- en points (pt), inches (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `h1 {height:100px}`

3. Les feuilles de style d'arrière-plan (background)

background-color

- définit la couleur de l'arrière-plan.
- couleur (par exemple en hexadécimal) ou transparent.
- `h1 {background-color:#000000}`

background-image

- définit l'image de l'arrière-plan.
- `url(fichier_image)` soit l'adresse (url) de l'image.
- `body {background-image:url(image.gif)}`

background-repeat

- définit la façon de répéter l'image d'arrière-plan.
- `repeat` ou `no-repeat` ou `repeat-x` (répétitions horizontales) ou `repeat-y` (répétitions verticales).
- `p {background-image:url(image.gif);background-repeat: repeat-4}`

background-attachment

- spécifie si l'image d'arrière-plan reste fixe avec les déplacements de l'écran.
- `scroll` ou `fixed`.
- `body {background-image:url(image.gif);background-attachment: fixed}`

background-position

- spécifie la position de l'image d'arrière-plan par rapport au coin supérieur gauche de la fenêtre.

- verticalement : top ou center ou bottom ou en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- horizontalement : left ou center ou right ou en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `body {background-image:url(img.gif);background-position: right top}`

background

- regroupe les différentes propriétés d'arrière-plan.
- `body {background:url(image.gif) fixed repeat}`

4. Les feuilles de style de marges ou de retrait (margin)

margin-top

- détermine la valeur de la marge supérieure.
- en unité de longueur ou auto.
- `img {margin-top:5px}`

margin-right

- détermine la valeur de la marge droite.
- en unité de longueur ou auto.
- `img {margin-right:5px}`

margin-bottom

- détermine la valeur de la marge inférieure.
- en unité de longueur ou auto.
- `img{margin-bottom:5px}`

margin-left

- détermine la valeur de la marge gauche. détermine la valeur de la marge gauche.
- en unité de longueur ou auto.
- `h3 {margin-left:5px}`

margin

- regroupe les différentes propriétés de la marge.
- `img {margin 5px}`

5. Les feuilles de style de bordure (border)

`border-top-width`

- donne l'épaisseur du bord supérieur.
- thin (fin) ou medium ou thick (gros) ou spécifié par l'auteur.
- `table{border-top-width:thin}`

`border-right-width`

- donne l'épaisseur du bord droit.
- thin ou medium ou thick ou spécifié par l'auteur.
- `table {border-right-width:medium}`

`border-bottom-width`

- donne l'épaisseur du bord inférieur.
- thin ou medium ou thick ou spécifié par l'auteur.
- `table{border-bottom-width:thick}`

`border-left-width`

- donne l'épaisseur du bord gauche.
- thin ou medium ou thick ou spécifié par l'auteur.
- `table {border-left-width:0.5cm}`

`border-width`

- regroupe les différentes propriétés de border-width.
- `table {border-width:thin}`

`border-color`

- détermine la couleur de la bordure.
- se décline aussi en border-top/right/bottom/left-color.
- `table {border-color:yellow}`

`border-style`

- détermine le style du trait de la bordure.
- se décline aussi en border-top/right/bottom/left-style.
- none ou solid ou double ou groove ou ridge ou inset ou outset.

- `table {border-style:none}`

`border`

- regroupe toutes les propriétés des bords.
- `table {border:0.2cm groove orange}`

6. Les feuilles de style d'enrobage (padding)

`padding-top`

- valeur de remplissage haut entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-top:3px}`

`padding-right`

- valeur de remplissage droit entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-right: 3px}`

`padding-bottom`

- valeur de remplissage bas entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-bottom: 3px}`

`padding-left`

- valeur de remplissage gauche entre l'élément et le bord.
- en points (pt), pouces (in), centimètres (cm), pixels (px) ou pourcentage (%).
- `td {padding-left: 3px}`

`padding`

- regroupe les différentes propriétés de remplissage.
- `td {padding:3mm 8mm}`

7. Les feuilles de style pour les listes

`list-style-type`

- détermine le type de puce ou de numérotation.

- disc (cercle plein) ou circle (cercle vide) ou square (carré).
- decimal ou lower-roman ou upper-roman ou lower-alpha ou upper-alpha.
- `ul {list-style-type:square}`

list-style-image

- permet de remplacer les puces par une image.
- `url(fichier_image)` ou `none`.
- `ul {list-style-image:url(image.gif)}`

list-style-position

- spécifie si les puces sont à l'intérieur ou à l'extérieur du texte.
- `inside` ou `outside`.
- `ul {list-style-position:inside}`

list-style

- regroupe toutes les propriétés de liste.
- `ul {list-style: outside url(dot.gif)}`

8. Les feuilles de style diverses

page-break-after

- ajoute un saut de page après l'élément. L'impression de la suite se fera sur une nouvelle page.
- `always` (toujours) ou `auto`.
- `p {page-break-after: always}`

page-break-before

- ajoute un saut de page avant l'élément. L'impression de la suite se fera sur une nouvelle page.
- `always` (toujours) ou `auto`.
- `p {page-break-after: always}`

visibility

- rend un élément visible ou caché.
- `visible` ou `hidden`.

z-index

- permet d'attribuer en ordre sur l'axe vertical. La position 0 est la plus en arrière. Les valeurs supérieures à 0

déterminent l'ordre. Ainsi 0 est plus en arrière que 1 et 1 est plus en arrière que 2...

- ``
- `<div style="position:absolute; top:100; left:100;z-index:1">Votre texte</div>`

9. Les pseudo-classes

`a:link`

- permet d'attribuer des feuilles de style au lien (non-visité).
- `a:link {text-decoration:none}`

`a:hover`

- permet d'attribuer des feuilles de style lors du survol du lien.
- `a:hover{color:red; text-transform:uppercase}`

`a:active`

- permet d'attribuer des feuilles de style au lien actif.
- `a:active{font-weight:bold; color:purple}`

`a:visited`

- permet d'attribuer des feuilles de style au lien visité.
- `a:visited {color: blue}`

`cursor`

- permet de changer la forme du curseur.
- `auto` ou `crosshair` (croix) ou `default` ou `hand` (main) ou `move` (déplacement) ou `text` (un texte éditables généralement I) `x-resize` (direction n, s, e, w) ou `wait` (sablier) ou `help` (point d'interrogation).
- `p{cursor:wait}`

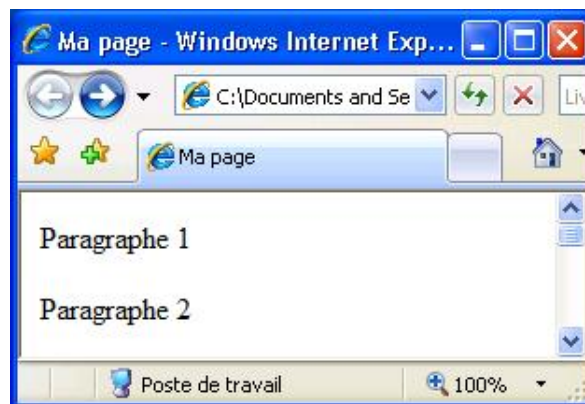
Les paragraphes de texte

Pour délimiter un paragraphe, une balise a été prévue soit `<p> ... </p>`.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<p>Paragraphe 1</p>
<p>Paragraphe 2</p>
texte
</body>
</html>
```

Résultat



Commentaires

- Par défaut, la balise `<p>` ajoute une ligne vide entre le texte de la balise et la suite du document.
- On a, pendant des années, présenté la balise de fermeture de la balise de paragraphe comme facultative. Avec les évolutions récentes des règles du Html réclamant plus de rigueur dans la conception du code, il est impératif de respecter scrupuleusement la fermeture des balises.
- Avec les balises de titre, la division en paragraphes contribue à la bonne structuration du contenu.
- Cette balise `<p>` est créée dans les éditeurs Html en enfonçant la touche [Entrée].

Le contrôle de passage à la ligne

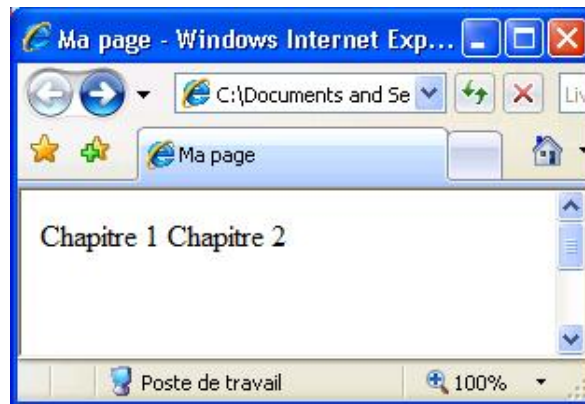
Le Html ignore les retours chariot ou passages à la ligne des éditeurs de texte.

Soit le code suivant, avec "Chapitre 1" et "Chapitre 2" encodés sur deux lignes :

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Chapitre 1
Chapitre 2
</body>
</html>
```

Résultat



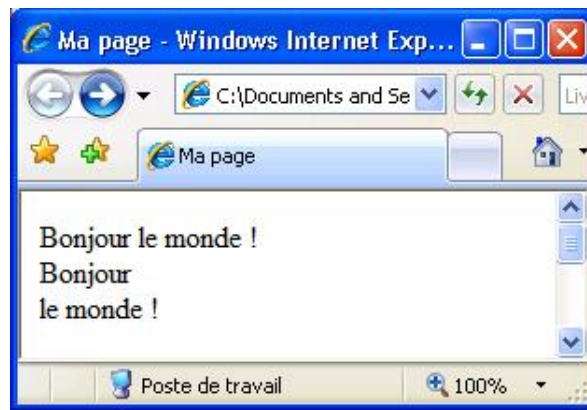
Commentaires

- Le passage à la ligne dans l'éditeur de texte est assimilé à un espace.
- Dans un document, il est cependant parfois nécessaire de passer à la ligne sans pour cela créer un paragraphe et la ligne vide qu'il ajoute : la balise
 est prévue à cet effet.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Bonjour
le monde !<br>
Bonjour<br>le monde !
</body>
</html>
```

Résultat



Commentaires

- La balise `
` appartient au groupe des balises pour lesquelles il n'y a pas de balises de fermeture.
- Ces balises sans balise de fermeture sont appelées des balises vides ou balises uniques.
- La balise `
` remplace les passages à la ligne de votre éditeur de texte.
- La balise `
` est créée dans les éditeurs Html en enfonçant les touches [Shift] et [Entrée].

Le formatage du texte

Classique des présentations de texte, il est possible de faire ressortir le texte en le mettant en gras ou en italique.

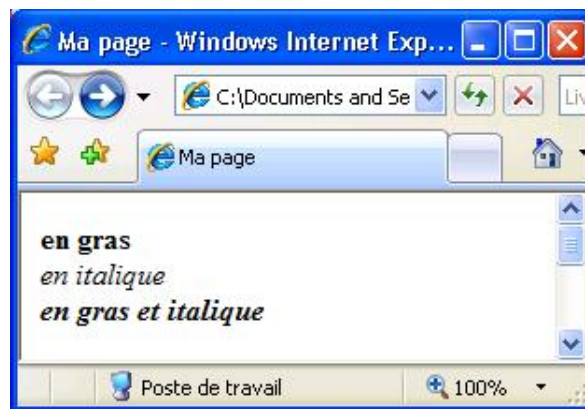
Pour mettre du texte en gras, on peut utiliser la balise ` ... ` (b pour bold).

Pour mettre du texte en italique, on utilise la balise `<i>... </i>` (i pour italic).

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<b>en gras</b><br>
<i>en italique</i><br>
<b><i>en gras et italique</i></b>
</body>
</html>
```

Résultat



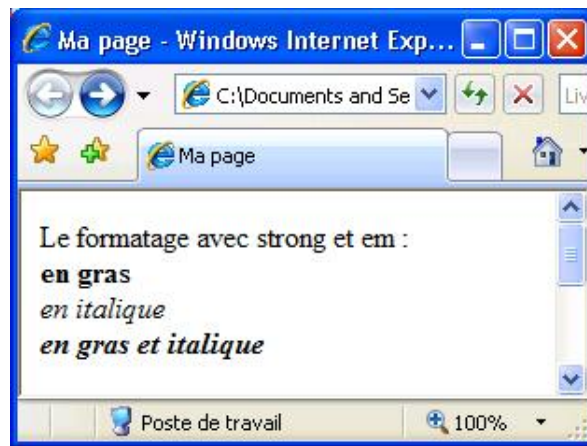
Commentaires

- Il est possible d'appliquer plusieurs balises à un élément de la page. Dans ce cas, veillez à respecter scrupuleusement l'ordre inverse pour les balises de fermeture. On parlera par la suite de balises correctement imbriquées.
- Il faut signaler les balises ` ... ` et ` ... ` qui correspondent respectivement à renforcer le texte et à mettre celui-ci en valeur. Dans les navigateurs modernes, la présentation est identique à mettre en gras et en italique.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Le formatage avec strong et em : <br>
<strong>en gras</strong><br>
<em>en italique</em><br>
<strong><em>en gras et italique</em></strong>
</body>
</html>
```

Résultat



Commentaire

- L'usage de ces balises est devenu rare (sauf dans certains éditeurs Html).

L'alignement

L'alignement du texte, ainsi que de tout élément du document comme par exemple les tableaux ou les images, est réalisé par l'attribut `align="type"` où "type" peut prendre la valeur :

- `left` pour un alignement à gauche (par défaut),
- `center` pour un alignement centré,
- `right` pour un alignement droite,
- `justify` pour un alignement justifié.

Comme vous rencontrez pour la première fois le terme d'attribut, il n'est pas inutile de rappeler qu'un attribut est un élément de code qui vient s'ajouter à une balise.

Exemple

Nous avons ajouté dans cet exemple l'attribut `align` à la balise de titre `<h2>`.

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<h2 align="left">Gauche</h2>
<h2 align="center">Centre</h2>
<h2 align="right">Droite</h2>
</body>
</html>
```

Résultat



Commentaire

- Il existait la balise `<center> ... </center>` pour centrer un élément (Html 3.2). Au demeurant fort pratique car plus concise que l'attribut `align="center"`, cette balise `<center>` n'est plus admise dans le standard Html 4.0 (*deprecated*). Son usage est donc déconseillé par le W3C.
- L'alignement justifié en Html donne parfois des résultats décevants sur le plan esthétique.

La taille, la couleur et la police

1. La taille du caractère

Pour modifier la taille des caractères, on peut utiliser la balise ` ... ` où x peut prendre, au choix, les valeurs de 1 à 7 ou de -3, -2, -1, 0, +1, +2, +3.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<font size="7">Bonjour</font><br>
<font size="+3">Bonjour</font>
</body>
</html>
```

Résultat



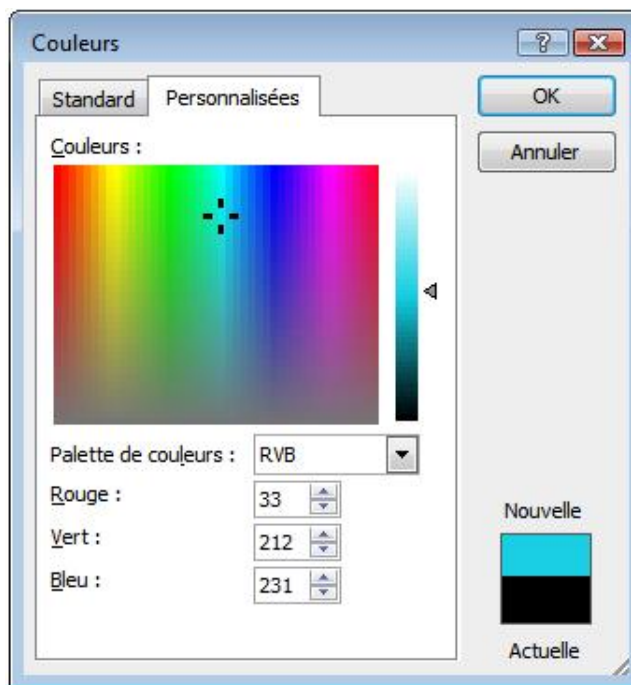
Commentaires

- Le Html 4.0 préconise l'usage des feuilles de style CSS (cf. Chapitre Les feuilles de style) pour la taille des caractères. Les possibilités offertes par ces dernières sont nettement plus variées que celles proposées par le Html..
- Une autre méthode pour modifier (de façon relative) la taille d'un texte est d'utiliser les balises `<big> ... </big>` pour agrandir le texte d'une taille et `<small> ... </small>` pour le diminuer d'une taille. On risque cependant d'alourdir inutilement le code source avec des écritures du style `<big><big><big>texte</big></big></big>`.

2. La couleur du texte

La définition de la couleur en langage Html est un peu particulière, sinon déroutante.

Chaque couleur est définie par une proportion de trois couleurs de base. C'est le système appelé RVB : Rouge, Vert et Bleu (RGB en anglais). Ce système est utilisé dans de nombreuses applications graphiques et même dans la fenêtre de couleurs de Windows comme ci-après.



On aurait pu définir tout simplement le rouge par exemple avec les repères 255,0,0 comme en RVB.

En Html, au lieu d'exprimer ces chiffres de 0 à 255, on a préféré les exprimer en valeur hexadécimale, base 16, soit de 00 à FF (soit deux positions au lieu de trois). Ce qui donne pour le rouge la valeur FF0000.

Comme cette base est peu utilisée, cette notation peut paraître assez sibylline. Elle présente quand même l'avantage de pouvoir exprimer en 6 chiffres quelques 16,7 millions de couleurs différentes.

Des applications graphiques comme Adobe Photoshop, Paint Shop Pro ou Gimp vous donnent l'équivalent hexadécimal des couleurs. En outre, de nombreux sites Web d'apprentissage sur le Html illustrent ces couleurs et leur équivalent en hexadécimal.

Par la suite, le W3C a introduit une liste de 16 noms de couleur qui peuvent également être utilisés soit black, maroon, green, olive, navy, purple, teal, gray, silver, red, lime, yellow, blue, fuchsia, aqua et white. Les firmes comme Microsoft et Netscape ont étendu cette liste mais certains noms de couleur ne sont pas communs aux deux firmes et l'usage du nom de couleurs (hors les 16 couleurs officielles) n'est pas dénué de risques. Les concepteurs de sites professionnels restent d'ailleurs fidèles à la notation hexadécimale (cf. chapitre Les images et arrière-plans pour plus de détails sur la notation des couleurs).

Pour changer la couleur d'un texte, on utilise la balise

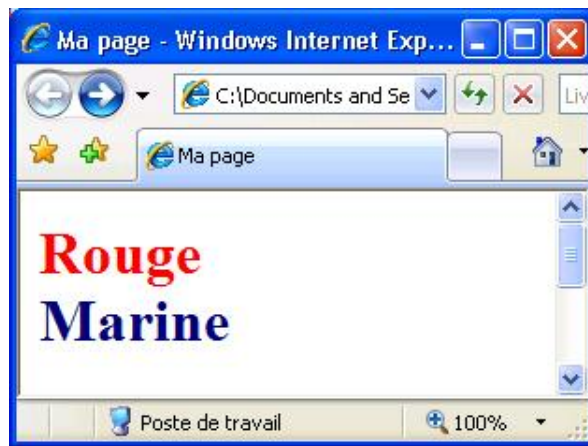
` ... ` pour la notation hexadécimale ou

` ... ` pour la notation par nom

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<h1>
<font color="#FF0000">Rouge</font><br>
<font color="navy">Marine</font>
</h1>
</body>
</html>
```

Résultat



Commentaire

- La couleur par défaut du texte de la page est le noir.
- Cette couleur par défaut peut être modifiée par l'attribut `text="#$$$$$"` appliqué à la balise `<body>`.

3. La police de caractères

Il est possible de changer la police définie par défaut dans le navigateur de l'utilisateur, généralement Times New Roman, par la balise

```
<font face="nom de la police"> ... </font>
```

Si la police de caractères n'est pas installée sur l'ordinateur de votre visiteur, son navigateur applique la police par défaut. Il est ainsi inutile d'utiliser des polices de caractères exotiques (comme vous invite à le faire Microsoft FrontPage), en conception de pages Web, qui n'existeront peut-être pas sur l'ordinateur de votre visiteur du bout du Web. Votre bel effet de présentation sera ainsi réduit à néant.

Il est même recommandé par précaution de proposer deux ou trois polices. Le navigateur regarde alors s'il peut afficher le texte dans la première police spécifiée, sinon il passe à la deuxième et ensuite à la troisième. Ce n'est que lorsqu'il a épuisé les deux ou trois propositions qu'il retient la police définie par défaut.

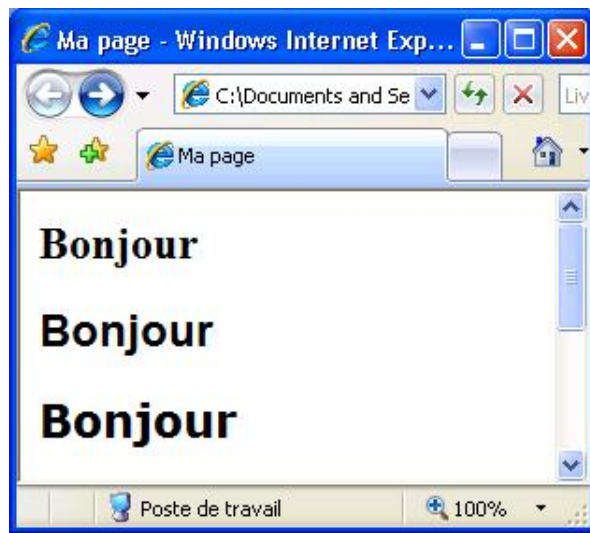
La balise se note alors, par exemple :

```
<font face="Verdana, Arial, Helvetica">
```

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<h2>Bonjour</h2>
<h2><font face="Arial">Bonjour</font></h2>
<h2><font face="Verdana">Bonjour</font></h2>
</body>
</html>
```

Résultat



Commentaire

- Dans le domaine de la police de caractères, les feuilles de style apportent aussi de nombreux développements (cf. Chapitre Les feuilles de style).

Les caractères spéciaux

Les anglo-saxons se retrouvent parfaitement à l'aise avec l'ASCII standard utilisé pour les documents Html. En effet, comme ils n'utilisent pas les accents, les 128 caractères de l'ASCII leur suffisent largement. Heureusement pour nous européens, dont beaucoup de langues sont d'essence latine, le Html reconnaît la norme ISO 8859-1 dite de l'alphabet latin-1. Mais ces caractères accentués ne seront affichés qu'au prix d'une "gymnastique" de caractères. On retrouve en Annexe Les caractères spéciaux, la liste (non exhaustive) de ces caractères spéciaux.

En voici quelques-uns :

le é s'encode avec é

le è s'encode avec è

le ê s'encode avec ˆ

le à s'encode avec `

le € s'encode avec €

l'espace insécable avec

Example

```
Elle passe l'été à rêvasser...
s'écrit en Html
Elle passe l'été à rêvasser...
```

Commentaire

- La syntaxe exacte comporte bien le point-virgule final. Même si Internet Explorer affiche correctement le caractère sans le point-virgule final, il n'en est pas de même avec d'autres navigateurs comme Firefox.



Le plus déroutant avec les navigateurs récents comme Internet Explorer 6 et 7 ainsi que Firefox est qu'ils acceptent certains caractères accentués (par exemple ; é) sans devoir passer par la notation académique (soit é). N'oubliez pas que le langage Html est un langage universel et que tout le monde n'utilise pas nécessairement une version récente des navigateurs.

Les commentaires et leur utilité

Le concepteur de pages Web souhaite parfois ajouter des commentaires dans une page de codes sans que ces annotations apparaissent à l'écran. Il dispose à cet effet de la balise de commentaire `<!-- commentaire -->`. Les commentaires sont ignorés par le navigateur lors de l'affichage de la page.

Ces balises de commentaires peuvent paraître anecdotiques mais elles se révèlent indispensables pour masquer parfois des lignes de code de langages plus évolués comme le JavaScript (cf. Chapitre Les autres langages du web).

Les autres balises de texte

Il existe de nombreuses balises de formatage de texte dont l'usage est occasionnel. Il importe cependant dans le cadre d'une formation au Html de les parcourir.

1. Les balises <strike> et <u>

Les balises <strike> et <u> permettent de tracer une ligne au travers ou en dessous d'un texte.

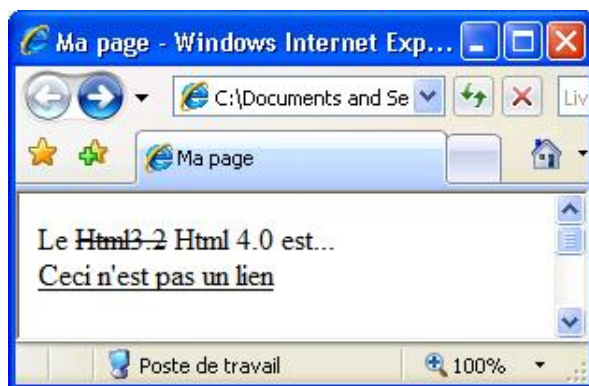
<strike> ... </strike> barre un texte pour indiquer, par exemple, qu'une version a été remplacée par une suivante.

<u> ... </u> souligne le texte.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Le <strike>Html3.2</strike> Html 4.0 est...<br>
<u>Ceci n'est pas un lien</u>
</body>
</html>
```

Résultat



Commentaire

- Le soulignement ne doit être utilisé dans la conception de pages Web qu'avec infiniment de circonspection car n'oubliez pas que sur le Web, le soulignement est la convention appliquée pour un lien hypertexte. Un soulignement pour mettre un texte en évidence ne peut que déconter le visiteur car il cliquera en vain espérant un lien.

2. Les balises <tt>, <code>, <kbd>, <samp>

Si vous encodez des codes informatiques ou d'autres caractères que vous désirez différencier du texte, vous pouvez les formater au moyen d'une police à pas fixe de style courrier. Bien que le résultat visuel soit identique, voici quelques balises :

<code> ... </code>

pour signaler du code informatique.

<kbd> ... </kbd>

pour signaler une saisie clavier.

<samp> ... </samp>

pour signaler un exemple.

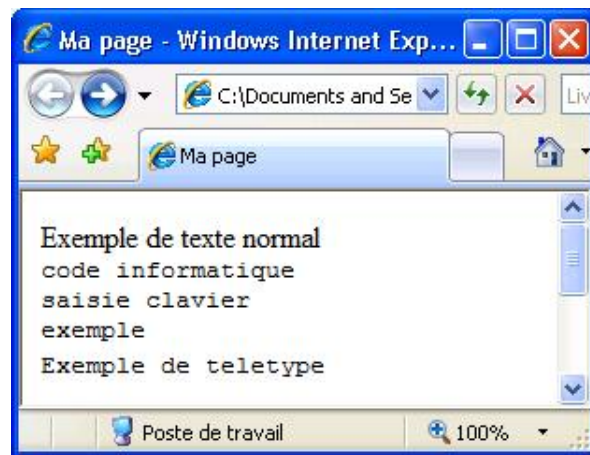
<tt> ... </tt>

la plus fréquente pour du texte de télétype.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Exemple de texte normal<br>
<code>code informatique</code><br>
<kbd>saisie clavier</kbd><br>
<samp>exemple</samp><br>
<tt>Exemple de teletype</tt>
</body>
</html>
```

Résultat



Ces balises ne sont pas d'un usage fréquent en Html.

3. Les balises <sup> et <sub>

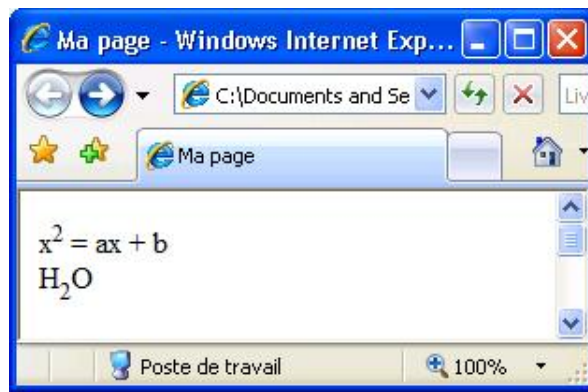
<sup> et <sub>

Pour mettre des chiffres ou du texte en exposant ou un indice, on dispose en Html de la balise ^{...} (exposant) et de _{...} (indice).

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
x<sup>2</sup> = ax + b<br>
H<sub>2</sub>O
</body>
</html>
```

Résultat



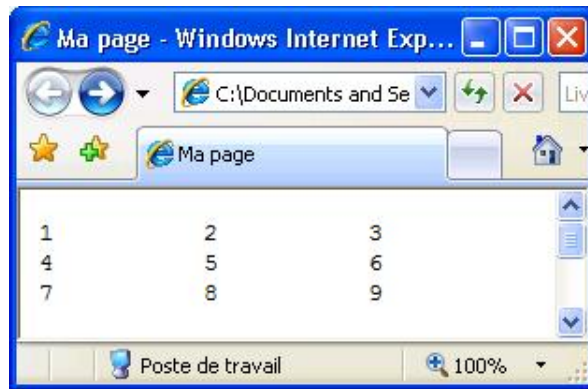
4. La balise <pre>

La balise <pre> ... </pre> affiche le texte préformaté (en police à pas fixe) soit avec les espaces, les retraits, les sauts de ligne tels qu'ils apparaissent à l'écran dans l'éditeur de texte.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<pre>
1          2          3
4          5          6
7          8          9
</pre>
</body>
</html>
```

Résultat



Pour la petite histoire du Html, la balise <pre> a connu son heure de gloire dans la première version du Html (1.0) car c'était à l'époque la seule façon d'encoder les tableaux (cf. Chapitre Les tableaux).

La balise <pre> fait plutôt partie maintenant des trucs et astuces d'écriture en Html, car elle permet d'apporter en quelques frappes de clavier des espaces verticaux de plusieurs lignes, plus importants que les paragraphes vides (<p>
</p>).

Une autre utilisation possible de cette balise est l'ajout de quelques lignes vides au code pour faire apparaître la barre de défilement verticale dans les captures d'écran de cet ouvrage.

Le code devient alors :

```
<html>
<head>
<title>Texte</title>
```

```
</head>
<body>
Code Html
<pre>

</pre>
</body>
</html>
```

5. La balise <blockquote>

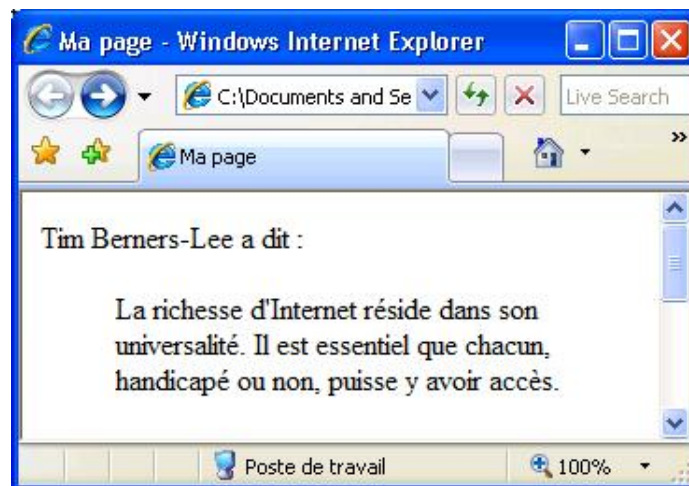
La balise <blockquote> ... </blockquote> a été conçue pour introduire une citation. Le texte de celle-ci s'affiche avec un léger retrait par rapport au texte normal.

Avant l'introduction des feuilles de style, cette balise a souvent été détournée de sa destination initiale pour introduire un retrait ou une marge gauche. Le W3C recommande d'en réserver l'usage pour les citations.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Tim Berners-Lee a dit :
<blockquote>
La richesse d'Internet réside dans son universalité. Il est essentiel
que chacun, handicapé ou non, puisse y avoir accès.
</blockquote>
</body>
</html>
```

Résultat



6. Les balises, encore et encore...

Par soucis du détail, voici encore d'autres balises de texte. Nous ne ferons que les parcourir rapidement car elles ne sont que (très) rarement utilisées.

Les balises dites sémantiques

Chacune de ces balises est destinée à spécifier la nature du contenu qu'elle encadre.

```
<acronym> ... </acronym>
```

Utilisé par les puristes du code pour signaler un acronyme. Pas d'effet visuel.

`<abbr> ... </abbr>`

Signale une abréviation. Pas d'effet visuel.

`<address> ... </address>`

Indique une adresse, généralement de courrier électronique. Met le texte en italique mais ne fait aucun lien vers cette adresse.

`<cite> ... </cite>`

Réservé aux citations d'un autre auteur. Met le texte de la citation en italique.

`<var> ... </var>`

Signale une variable. Met le texte en italique.

La balise `<nobr>`

La balise `<nobr> ... </nobr>` demande au navigateur de ne pas couper le mot ou une expression et donc de l'afficher sur une même ligne. Cette balise ne fait plus partie du Html 4.

Les marges de la fenêtre

Les navigateurs comme Internet Explorer et Firefox ménagent tous deux, par défaut, une marge horizontale et verticale entre les bords de la fenêtre et le contenu de la page. Il existe des attributs de la balise `<body>` qui permettent de gérer l'espace souhaité à cet endroit.

On peut contrôler les marges par défaut en ajoutant les attributs `leftmargin="x"` pour la marge horizontale et `topmargin="x"` pour la marge supérieure où `x` est exprimé en pixels.

Ainsi, pour un document qui s'afficherait sans aucune marge et qui serait compatible pour Internet Explorer et Firefox, la balise `<body>` serait : `<body leftmargin="0" topmargin="0">`.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body leftmargin="0" topmargin="0">
Les navigateurs comme Internet Explorer et Firefox ménagent tous deux, par défaut,
une marge horizontale et verticale entre les bords de la fenêtre et le contenu de
la page. Il existe des attributs de la balise body qui permettent de gérer
l'espace souhaité à cet endroit.
</body>
</html>
```

Résultat



Commentaire

- Ces attributs n'ont jamais fait partie d'une quelconque norme Html. Elles datent de l'époque, un peu désordonnée, où les navigateurs inventaient des balises sans se soucier des standards du W3C. Utilisez les feuilles de style CSS pour définir les marges de la page.

Les listes

Les listes numérotées ou à puces ne peuvent être absentes dans la présentation d'un document Html.

1. Les listes numérotées

Démarche classique en Html, on détermine d'abord que le navigateur doit mettre en place une liste, ici numérotée ou ordonnée : soit la balise ` ... `. À l'intérieur de ces balises, on détermine ensuite les éléments de la liste soit ` ... `.

Cette façon de procéder est la même dans d'autres éléments du Html comme les tableaux, les cadres, les formulaires, etc.

La balise `` comporte des attributs :

- `type="x"` où x correspond à :

A pour des majuscules,

a pour des minuscules,

I pour des chiffres romains majuscules,

i pour des chiffres romains minuscules,

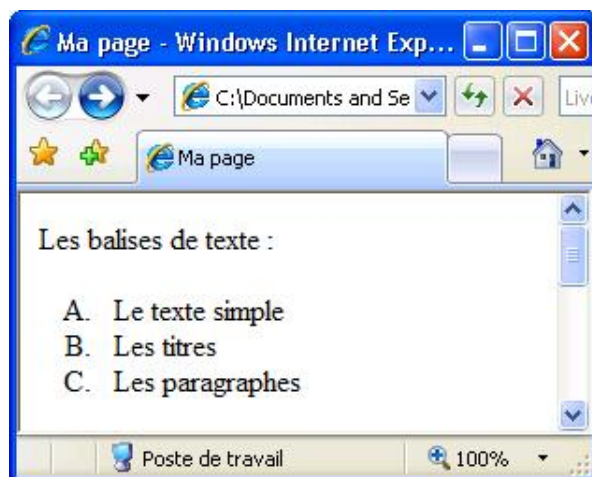
1 (par défaut) pour des nombres.

- `start="x"` où x représente la valeur (en nombre) de début de la numérotation.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Les balises de texte :
<ol type="A">
<li>Le texte simple</li>
<li>Les titres</li>
<li>Les paragraphes</li>
</ol>
</body>
</html>
```

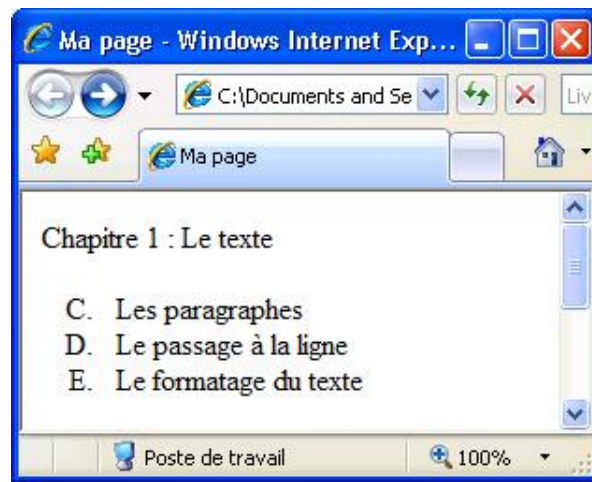
Résultat



Un autre exemple avec un début de numérotation différent

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Chapitre 1 : Le texte
<ol type="A" start="3">
<li>Les paragraphes</li>
<li>Le passage &agrave; la ligne</li>
<li>Le formatage du texte</li>
</ol>
</body>
</html>
```

Résultat



Commentaires

- La balise `` comme pour les autres listes introduit un léger retrait du texte.
- N'oubliez pas la balise de fermeture de la liste `` car dans ce cas le retrait continue à être appliqué par le navigateur.
- Il y a quelques années, la balise de fermeture de liste `` était présentée comme facultative. Avec la rigueur réclamée par le W3C dans l'écriture du code Html, cette balise de fermeture est bien obligatoire.
- Les feuilles de style (cf. Chapitre Les feuilles de style) renouvellent un peu la présentation des listes.

2. Les listes à puces

On informe d'abord le navigateur qu'il a à mettre en place une liste à puces ou non ordonnée, soit la balise ` ... `. À l'intérieur de ces balises, on détermine ensuite les éléments de la liste soit ` ... `.

La balise `` comporte un attribut `type="forme"` où forme correspond au type de puces.

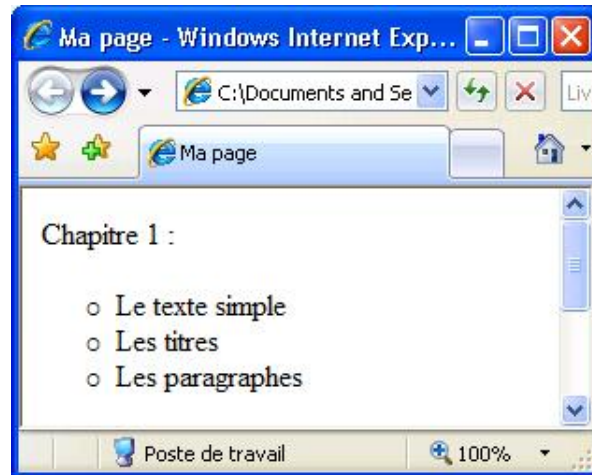
En Html, vous avez le choix entre trois formes de puces :

- `disc` pour une puce ronde et pleine,
- `circle` pour une puce ronde et vide
- `square` pour une puce carrée.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Chapitre 1 :
<ul type="circle">
<li>Le texte simple</li>
<li>Les titres</li>
<li>Les paragraphes</li>
</ul>
</body>
</html>
```

Résultat



Commentaires

- La balise `` comme pour les autres listes introduit un léger retrait du texte.
- N'oubliez pas la balise de fermeture de la liste ``, sinon le retrait continue à être appliqué par le navigateur.
- La balise de fermeture `` est bien obligatoire.
- Pour une présentation plus esthétique que les puces usuelles du Html, l'usage voulait que l'on passe par des images et des puces.

Exemple (avec les bordures du tableau visibles)

Chapitre 1 :
▶ Le texte simple
▶ Les titres
▶ Les paragraphes
▶ Le formatage du texte

Les feuilles de style CSS renouvellent la présentation des listes à puces en permettant notamment d'inclure n'importe quelle image en lieu et place des puces.

3. Les listes imbriquées

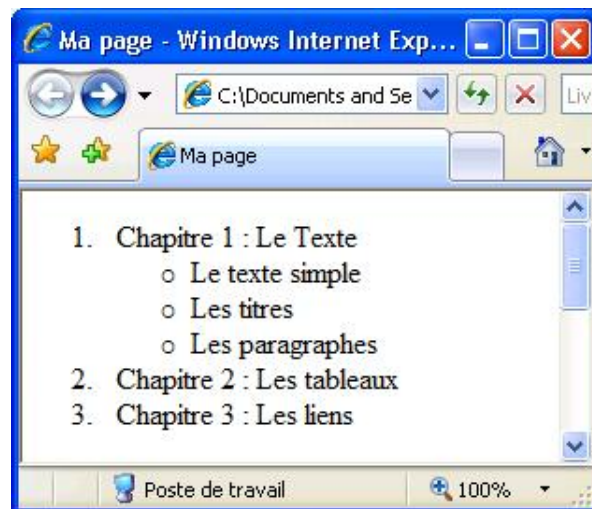
Rien ne s'oppose à élaborer des listes numérotées et des listes à puces imbriquées (ou inversement). Le codage est simplement un peu plus complexe.

Soit une liste numérotée dans laquelle s'imbrique une liste à puces.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<ol>
<li>Chapitre 1 : Le Texte
    <ul type="circle">
    <li>Le texte simple</li>
    <li>Les titres</li>
    <li>Les paragraphes</li>
    </ul></li>
<li>Chapitre 2 : Les tableaux</li>
<li>Chapitre 3 : Les liens</li>
</ol>
</body>
</html>
```

Résultat



4. Les listes de définition

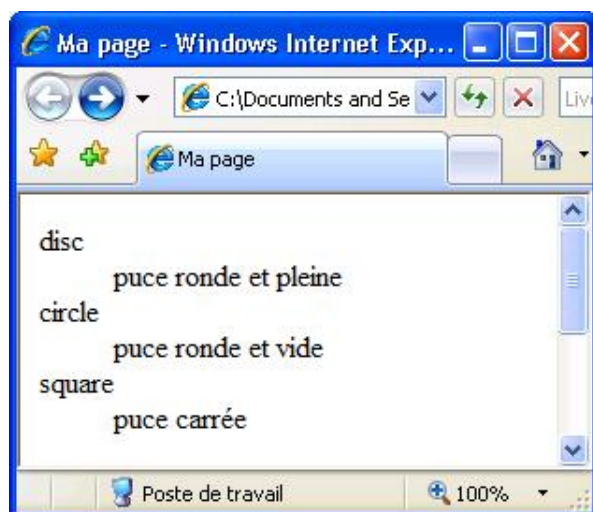
Le Html propose une dernière sorte de liste, particulièrement utile pour présenter par exemple des glossaires. Dans cette liste, dite de définition, il y a deux éléments : un terme et sa description.

La liste se construit d'abord par la déclaration d'une liste de définition `<dl> ... </dl>`, Entre ces balises, on introduit les balises qui reprennent ce qu'on a appelé un terme soit `<dt> ... </dt>` et puis les balises pour la description `<dd> ... </dd>`.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<dl>
<dt>disc</dt>
<dd>puce ronde et pleine</dd>
<dt>circle</dt>
<dd>puce ronde et vide</dd>
<dt>square</dt>
<dd>puce carrée</dd>
</dl>
</body>
```

Résultat



Commentaire

- La balise <dd> introduit un retrait.

Les séparateurs horizontaux

Le trait horizontal est un outil fort pratique pour structurer le contenu. Le Html incorpore un élément graphique pour le réaliser. C'est la balise `<hr>`.

Cette balise comporte quelques attributs :

`size="x"`

pour déterminer la hauteur du trait en pixels.

`width="x" OU width="x%"`

pour la largeur du trait en pixels ou en pourcentage.

`align="left" OU "center" OU "right"`

pour l'alignement du trait.

`noshade`

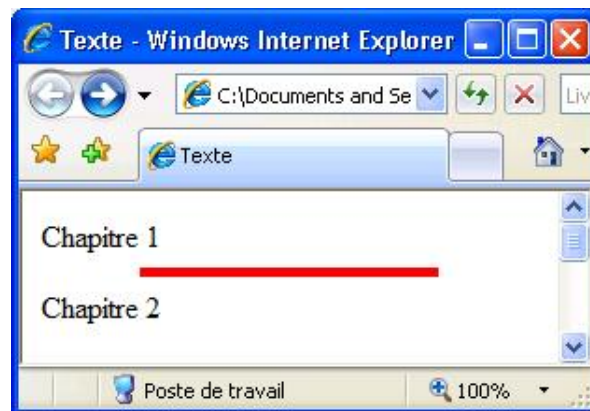
pour créer un trait plein sans effet d'ombrage.

et dans Internet Explorer seulement `color="couleur"`.

Exemple

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
Chapitre 1
<hr size="5" width="60%" align="center" noshade color="red">
Chapitre 2
</body>
</html>
```

Résultat



Commentaires :

- La balise `<hr>` ne comporte pas de balise de fermeture.
- Les concepteurs préfèrent remplacer ce trait, assez simpliste, par une image, graphiquement plus élaborée.
- L'attribut `color` utilisé dans l'exemple, ne fait pas partie de la norme Html 4.0.

L'utilité dans la conception des sites Web

Pour les habitués des traitements de texte, les tableaux sont surtout perçus comme une façon esthétique de présenter des données. Dans la conception des pages Web, les tableaux sont souvent incontournables pour découper toute votre page en zones facilement gérables. On parlera dans ce cas de tableaux de présentation.

Il n'est pas rare qu'une page Web professionnelle soit un enchevêtrement de tableaux imbriqués.

➤ Cependant, ces dernières années, de nombreux designers Web utilisent les feuilles de style au lieu des tableaux pour positionner les éléments d'une page. Cela entraîne non seulement un vrai respect des spécifications du W3C, mais surtout un code source plus concis et plus lisible.

La structure des tableaux

1. La création d'un tableau

Un tableau (<table>) comporte des lignes (<tr>). Ces lignes comprennent plusieurs cellules (<td>). Réalisons en Html le tableau suivant :

1	2
3	4

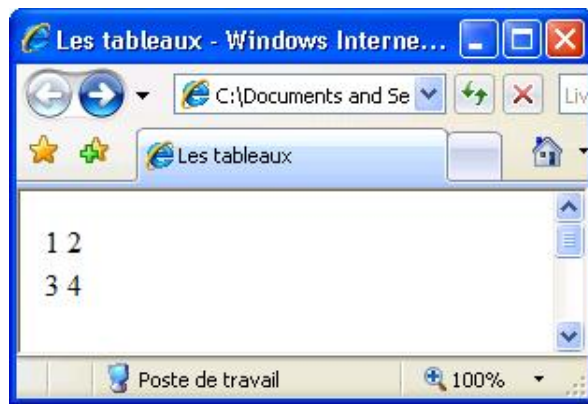
Dans la logique du Html, le tableau se construit ainsi :

- On déclare que l'on ouvre un tableau avec la balise <table>.
- Ce tableau comporte une première ligne soit <tr>.
- Cette ligne comporte une première cellule <td>.
- Soit 1 comme contenu de la cellule.
- Fermeture de cette première cellule </td>.
- Vient une seconde cellule <td>2</td>.
- Fin de la première ligne </tr>.
- On passe à la ligne suivante soit à nouveau <tr>.
- Cette ligne comporte les cellules <td>3</td> et <td>4</td>.
- Fin de la seconde ligne </tr>.
- Et fin du tableau avec la balise de fermeture </table>.

Exemple

```
<html>
<head>
<title>Les tableaux</title>
</head>
<body>
<table>
<tr>
<td>1</td><td>2</td>
</tr>
<tr>
<td>3</td><td>4</td>
</tr>
</table>
</body>
</html>
```

Résultat



Commentaire

- La balise `<table>` possède, heureusement, de nombreux attributs pour améliorer la présentation des tableaux.

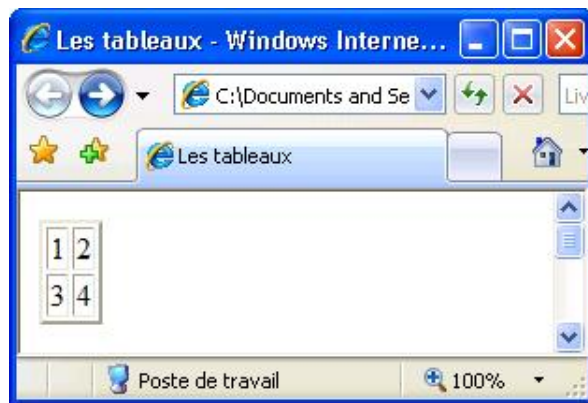
2. Les bordures du tableau

Il est possible d'ajouter une bordure par l'attribut `border="x"` où `x` est un nombre exprimé en pixels.

Exemple

```
<body>
<table border="2">
<tr>
<td>1</td><td>2</td>
</tr>
<tr>
<td>3</td><td>4</td>
</tr>
</table>
</body>
```

Résultat



Commentaire

- Il est possible de donner une couleur à la bordure par l'attribut `bordercolor=couleur` (cf. section Les cellules des tableaux - La couleur de bordure d'une cellule de ce chapitre).
- Les feuilles de style CSS permettent de renouveler les bordures de tableau.

3. L'alignement du tableau

Il est possible d'aligner le tableau à gauche (par défaut), au centre et à droite par l'attribut `align="left"` ou `"center"`

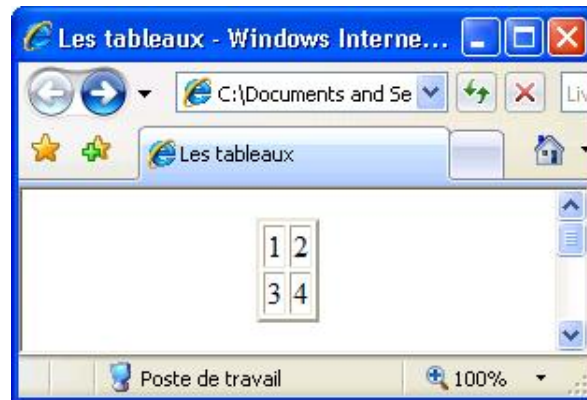
ou "right".

Pour centrer le tableau, le code devient :

Exemple

```
<body>
<table align="center" border="2">
<tr>
<td>1</td><td>2</td>
</tr>
<tr>
<td>3</td><td>4</td>
</tr>
</table>
</body>
```

Résultat



4. L'espace entre les cellules du tableau

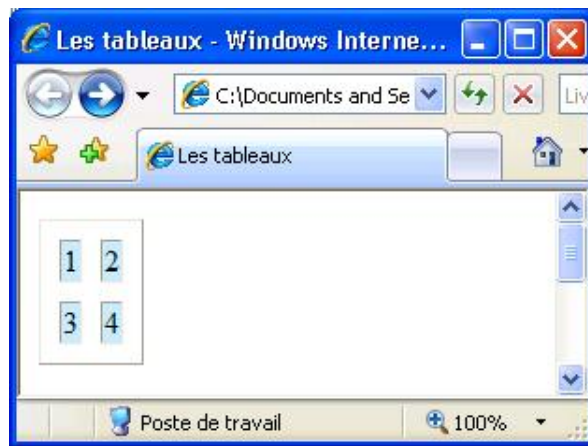
L'espace entre les cellules du tableau est défini par défaut à 2 pixels. L'attribut `cellspacing="x"` permet de modifier cet espacement entre les cellules.

Pour un espace entre les cellules de 20 pixels, vous pouvez utiliser le code :

Exemple

```
<body>
<table cellspacing="20">
<tr>
<td>1</td><td>2</td>
</tr>
<tr>
<td>3</td><td>4</td>
</tr>
</table>
</body>
```

Résultat



Commentaire

- La couleur de fond des cellules a été ajoutée pour mieux visualiser l'espacement entre les cellules. Le code de la couleur de fond est absent du code de l'exemple car il ne sera abordé qu'ultérieurement dans le livre (cf. Les cellules des tableaux - La couleur de fond d'une cellule du présent chapitre).

5. La marge à l'intérieur des cellules

La marge à l'intérieur des cellules est l'espace qui sépare les bords de la cellule et le contenu de celle-ci. L'attribut `cellpadding="x"` permet de modifier cet espacement entre les bords de la cellule.

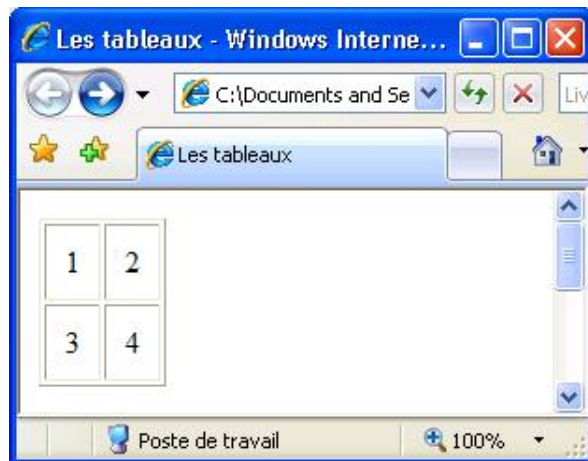
Par défaut cette marge est de 1 pixel. Ce qui est peu et généralement très disgracieux car, dans ce cas, le texte est littéralement collé à la bordure. Préférez une marge de 3 ou 5 pixels.

À titre d'exemple, prenez une marge intérieure de 10 pixels.

Exemple

```
<body>
<table cellpadding="10" border="1">
<tr>
<td>1</td><td>2</td>
</tr>
<tr>
<td>3</td><td>4</td>
</tr>
</table>
</body>
```

Résultat



Commentaire

- La marge est apportée tout autour du contenu de la cellule.

6. La taille du tableau

La largeur du tableau est déterminée par l'attribut `width="x"` ou `"x%"` où `x` est exprimé en pixels et `x%` correspond à un pourcentage de la fenêtre.

Il existe aussi un attribut `height="x"` ou `"x%"` pour définir la hauteur du tableau mais son emploi est moins fréquent.

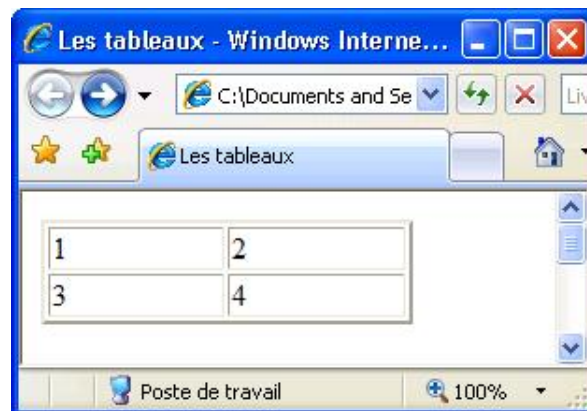
Il faut cependant noter que cet attribut `height` n'appartient pas au standard `Html 4.0`. Il est cependant correctement interprété par les navigateurs.

Soit un tableau d'une largeur de 75 % de la fenêtre.

Exemple

```
<body>
<table width="75%" border="1">
<tr>
<td>1</td><td>2</td>
</tr>
<tr>
<td>3</td><td>4</td>
</tr>
</table>
</body>
```

Résultat



Les cellules des tableaux

Les cellules peuvent contenir tous les éléments définis par le Html, soit du texte, des images, des liens, des arrière-plans et même... des tableaux.

1. La largeur des cellules

Par défaut, le navigateur adapte la largeur des cellules selon leur contenu.

Exemple d'un tableau de 3 colonnes égales mais **sans spécifications particulières** :

cellule 1	cel. 2	3
cellule 4	cel. 5	6

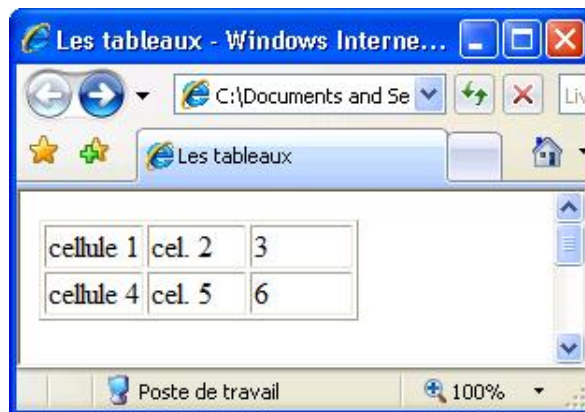
L'attribut `width="x"` ou `"x%"`, appliqué cette fois à la balise `<td>`, permet de définir la largeur de la cellule soit en valeur absolue (en pixels) soit en valeur relative (en %) de la taille du tableau.

Appliqué au tableau précédent, on obtiendra des colonnes égales par le code suivant :

Exemple

```
<body>
<table border="1">
<tr>
<td width="33%">cellule 1</td>
<td width="33%">cel. 2</td>
<td width="34%">3</td>
</tr>
<tr>
<td>cellule 4</td>
<td>cel. 5</td>
<td>6</td>
</tr>
</table>
</body>
```

Résultat



Commentaire

- La première ligne du tableau détermine l'agencement du tableau. Il n'est donc pas indispensable de reprendre le code pour toutes les lignes (comme le font la plupart des éditeurs Html).

2. L'alignement horizontal

Chaque cellule du tableau est indépendante. Il est donc possible d'attribuer un alignement distinct à chaque cellule.

L'attribut align="left" ou "center" ou "right" permet d'aligner (horizontalement) le contenu de la cellule à gauche (par défaut), au centre ou à droite.

Exemple

```
<body>
<table border="1" width="75%">
<tr>
<td width="33%" align="left">1</td>
<td width="33%" align="center">2</td>
<td width="34%" align="right">3</td>
</tr>
<tr>
<td align="right">4</td>
<td align="center">5</td>
<td align="left">6</td>
</tr>
</table>
</body>
```

Résultat



Commentaires

- Il est possible d'aligner le contenu de toutes les cellules d'une ligne en insérant l'attribut align dans la balise <tr> appropriée. La valeur par défaut est left.
- L'alignement spécifié dans la cellule <td> a priorité sur l'alignement de la ligne <tr>.

3. L'alignement vertical

L'attribut valign="type" permet d'aligner verticalement le contenu d'une cellule ou d'une ligne. Les valeurs de "type" offrent les possibilités suivantes :

- **top** pour haut.
- **middle** pour centre (par défaut).
- **bottom** pour bas.
- **baseline** pour la ligne de base de la première ligne de texte.



Attention ! La valeur par défaut (middle) risque de vous apporter des surprises assez inesthétiques lors de l'élaboration de votre tableau...

1	xxxxx xxxxx xxxxx xxxxx xxxxx
2	xxxxx xxxxx
3	xxxxx xxxxx xxxxx xxxxx xxxxx xxxxx xxxxx

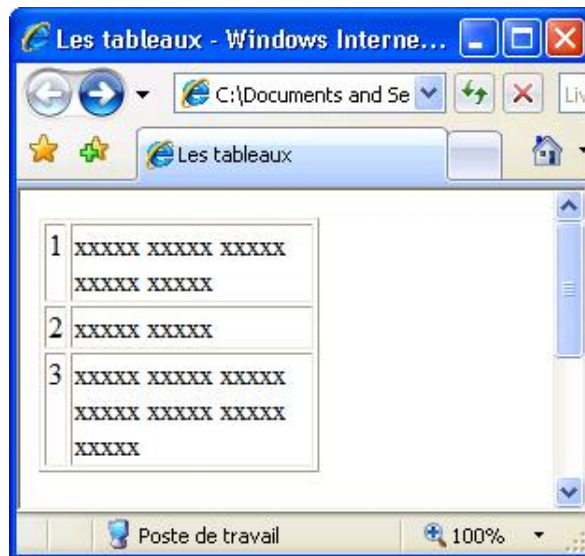
Commentaires

- Il est donc prudent, pour avoir de belles présentations de tableaux, de spécifier pour chaque ligne, les attributs align et valign. Soit `<tr align="left" valign="top">`.
- L'alignement spécifié dans la cellule `<td>` a priorité sur l'alignement de la ligne `<tr>`.

Exemple

```
<body>
<table border="1" width="150">
<tr align="left" valign="top">
<td>1</td><td>xxxxx xxxxx xxxxx xxxxx xxxxx </td>
</tr>
<tr align="left" valign="top">
<td>2</td><td> xxxxx xxxxx </td>
</tr>
<tr align="left" valign="top">
<td>3</td><td> xxxxx xxxxx xxxxx xxxxx xxxxx xxxxx xxxxx </td>
</tr>
</table>
</body>
```

Résultat



4. La couleur de fond d'une cellule

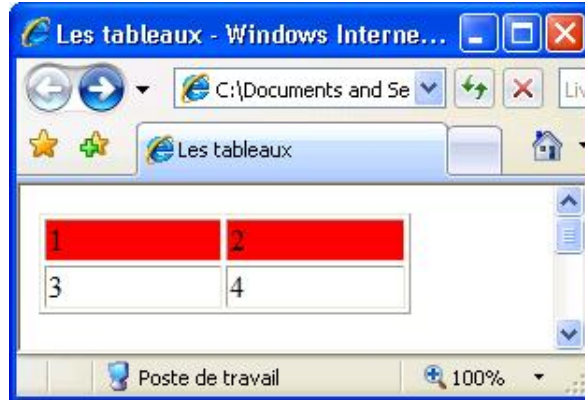
Il est souvent fort utile de pouvoir modifier la couleur d'une cellule, d'une ligne ou d'un tableau pour mettre en évidence certaines données.

L'attribut `bgcolor="#$$$$$"` ou `bgcolor="nom"`, où `$$$$$` est la notation hexadécimale des couleurs (cf. Chapitre Images et arrière-plans) et `"nom"` le nom (reconnu par le navigateur) de la couleur, permet de changer la couleur de fond lorsqu'il est appliqué à une cellule (`<td>`), à une ligne (`<tr>`) ou à un tableau (`<table>`).

Exemple

```
<body>
<table width="75%" border="1">
<tr bgcolor="#FF0000">
<td width="50%">1</td>
<td width="50%">2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</table>
</body>
```

Résultat



Commentaire

- La première ligne comprend un fond rouge.

5. La couleur de bordure d'une cellule

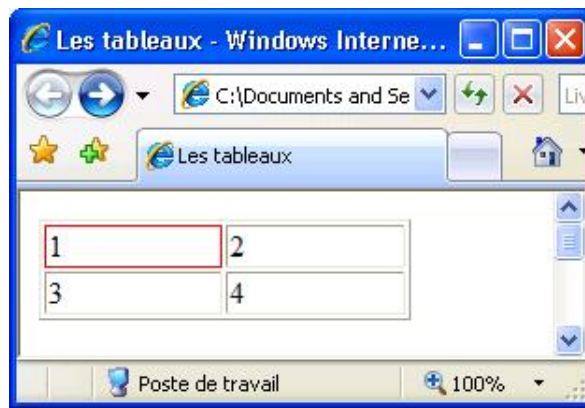
Il est possible d'entourer une cellule particulière avec un bord de couleur.

L'attribut `bordercolor="#$$$$$$"` ou `bordercolor="nom"`, où `$$$$$$` est la notation hexadécimale des couleurs et "nom" le nom (reconnu par le navigateur) de la couleur, permet de donner une couleur à la bordure de la cellule.

Exemple

```
<body>
<table width="75%" border="1">
<tr>
<td width="50%" bordercolor="red">1</td>
<td width="50%">2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</table>
</body>
```

Résultat



Cet attribut bordercolor appliqué à la balise <td> ne fonctionne pas sous Firefox.

Cet attribut peut bien entendu s'appliquer aussi aux cellules du tableau <table> et aux cellules de ligne <tr>.

Exemple

```
<body>
<table width="75%" border="1" bordercolor="red">
<tr>
<td width="50%">1</td>
<td width="50%">2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</table>
</body>
```

Résultat



Avec Internet Explorer, le tableau est quadrillé d'un double trait de couleur rouge.



Sous Firefox, seul le contour du tableau est entouré d'un unique trait rouge.

Cet exemple illustre les (légères) différences d'affichage des pages selon les navigateurs. Dans le cas présent, l'attribut bordercolor est un attribut propriétaire de Microsoft et n'a jamais fait partie d'une norme du W3C. Ce qui explique sa prise en compte très partielle par Firefox.

Pour obtenir une apparence identique sous Internet Explorer et Firefox, il faut passer par les feuilles de style CSS.

Commentaire

- Il est possible de n'avoir qu'un seul trait (au lieu de deux traits avec l'espacement de cellules par défaut) en ajoutant l'attribut cellspacing="0". Le code devient :

```
<body>
<table width="75%" border="1" bordercolor="red" cellspacing="0">
<tr>
<td width="50%">1</td>
<td width="50%">2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</table>
</body>
```



Cette astuce ne fonctionne que sous Internet Explorer et fournit à nouveau un résultat différent sous Firefox.

6. La cellule vide

Lorsqu'une cellule de tableau est vide, le résultat affiché n'est pas particulièrement élégant.

Exemple

```
<body>
<table width="75%" border="1">
<tr>
<td width="50%">1</td>
<td width="50%"></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</table>
</body>
```

Résultat



Il suffit d'ajouter un espace insécable entre les balises de la cellule pour retrouver un affichage correct.

Exemple

```
<body>
<table width="75%" border="1">
<tr>
<td width="50%">1</td>
<td width="50%">&nbsp;</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</table>
</body>
```

Résultat



La fusion des cellules

Il est possible de fusionner horizontalement ou verticalement des cellules.

Soit un tableau de départ de 2 lignes et 3 colonnes.

Exemple

```
<body>
<table width="80%" border="1">
<tr>
<td width="33%">1</td>
<td width="33%">2</td>
<td width="34%">3</td>
</tr>
<tr>
<td width="33%">1</td>
<td width="33%">2</td>
<td width="34%">3</td>
</tr>
</table>
</body>
```

Résultat



1. La fusion de colonnes

Pour fusionner des colonnes, le Html dispose de l'attribut de cellule `colspan="x"` où x correspond au nombre de colonnes que l'on souhaite fusionner horizontalement.

Exemple

```
<body>
<table width="80%" border="1">
<tr>
<td colspan="3" align="center">Titre</td>
</tr>
<tr>
<td width="33%">1</td>
<td width="33%">2</td>
<td width="34%">3</td>
</tr>
</table>
</body>
```

Résultat



Commentaires

- La fusion des trois colonnes de la première ligne permet par exemple d'y indiquer un titre.
- Il n'y a bien entendu, dans le cas présent, plus qu'une seule cellule dans la première ligne.

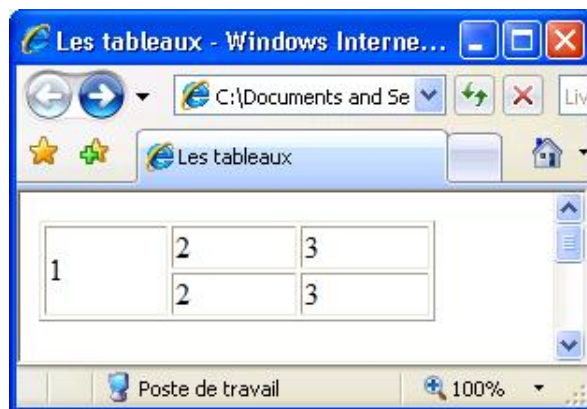
2. La fusion de lignes

Pour fusionner des lignes, le Html dispose de l'attribut de cellule `rowspan="x"` où `x` correspond au nombre des cellules que l'on souhaite fusionner verticalement.

Exemple

```
<body>
<table width="80%" border="1">
<tr>
<td width="33%" rowspan=2>1</td>
<td width="33%">2</td>
<td width="34%">3</td>
</tr>
<tr>
<td width="33%">2</td>
<td width="34%">3</td>
</tr>
</table>
</body>
```

Résultat



Commentaires

- La première colonne des deux lignes du tableau est fusionnée.
- Il n'y a bien entendu, dans le cas présent, plus que deux cellules dans la seconde ligne du tableau.

Le titre ou la légende de tableau

La balise `<caption>` permet d'associer un titre ou une légende de tableau. Par l'attribut `align="top"`, ce commentaire apparaît au-dessus du tableau et apparaît ainsi comme un titre de celui-ci. Avec l'attribut `align="bottom"`, le texte de la balise `<caption>` s'affiche en dessous du tableau fonctionnant comme une légende de celui-ci.

La balise `<caption>` doit être placée après la balise `<table>` ouvrante et ne peut apparaître qu'une fois dans le tableau.

Exemple

```
<html>
<head>
<title>Les tableaux</title>
</head>
<body>
<table width="75%" border="1">
<caption align="bottom">La légende de tableau</caption>
<tr>
<td>1</td><td>2</td>
</tr>
<tr>
<td>3</td><td>4</td>
</tr>
</table>
</body>
</html>
```

Résultat



Spécifier un en-tête de ligne ou de colonnes

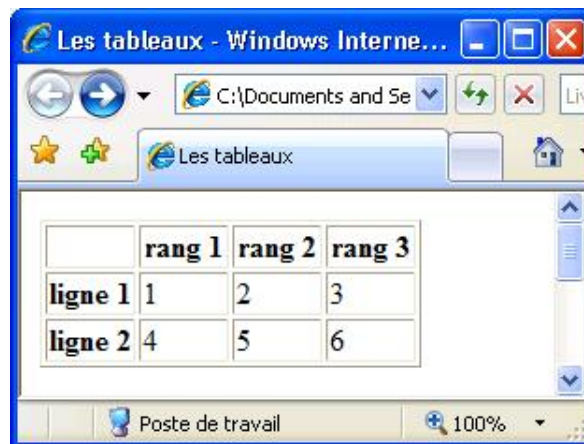
Dans un tableau de données, les balises <th> ... </th> spécifient les en-têtes de colonnes ou de ligne. Les navigateurs les affichent par défaut en gras et avec un alignement centré.

Cette balise <th> fait partie des premières balises du Html. Elle serait tombée en désuétude si elle n'était d'une grande utilité pour l'accessibilité des sites Web. En effet, elle permet aux utilisateurs de lecteurs d'écran de connaître le titre de la ligne ou de la colonne dans laquelle ils se trouvent. Les personnes aveugles ont une lecture linéaire des tableaux. Aussi, le marquage des en-têtes de colonnes constitue-t-il un point de repère essentiel.

Exemple

```
<body>
<table border="1">
<tr>
<th width="25%">&nbsp;</th>
<th width="25%">rang 1</th>
<th width="25%">rang 2</th>
<th width="25%">rang 3</th>
</tr>
<tr>
<th width="25%">ligne 1</th>
<td width="25%">1</td>
<td width="25%">2</td>
<td width="25%">3</td>
</tr>
<tr>
<th width="25%">ligne 2</th>
<td width="25%">4</td>
<td width="25%">5</td>
<td width="25%">6</td>
</tr>
</table>
</body>
```

Résultat



	rang 1	rang 2	rang 3
ligne 1	1	2	3
ligne 2	4	5	6

Préserver d'autres bordures

Avec l'attribut `border` de la balise `<table>`, la bordure encadre toutes les cellules du tableau. Avec la spécification 4.0 du HTML, il est maintenant possible de choisir les côtés auxquels on pourra affecter une bordure.

Pour spécifier les **bordures extérieures** à afficher, on utilise dans la balise `<table>`, après l'attribut `border`, obligatoire, l'attribut `frame="type"` où `type` prend les valeurs suivantes :

void

Pour aucune bordure.

above

Pour une bordure unique sur le bord supérieur.

below

Pour une bordure unique sur le bord inférieur.

hsides

Pour des bordures sur les bords supérieurs et inférieurs.

vsides

Pour des bordures sur les côtés gauche et droit.

rhs

Pour des bordures sur le côté droit.

lhs

Pour des bordures sur le côté gauche.

box ou **border**

Pour des bordures sur tous les côtés.

Pour spécifier les bordures intérieures à afficher, on utilise dans la balise `<table>`, après l'attribut `border` obligatoire, l'attribut `rules="type"` où `type` prend les valeurs :

none

Pour aucun trait intérieur.

rows

Pour un trait horizontal entre chaque ligne.

cols

Pour un trait vertical entre chaque colonne.

all

Pour un trait entre chaque ligne et chaque colonne.

groups

Pour un trait entre les groupes ou sections.

Exemple d'un tableau avec uniquement les bords supérieurs et inférieurs sans les bordures intérieures :

```
<body>
<table border="1" frame="hsides" rules="none" width="75%" cellspacing="0">
<tr>
<td width="50%">1</td>
<td width="50%">2</td>
</tr>
<tr>
<td width="50%">3</td>
<td width="50%">4</td>
</tr>
</table>
</body>
```

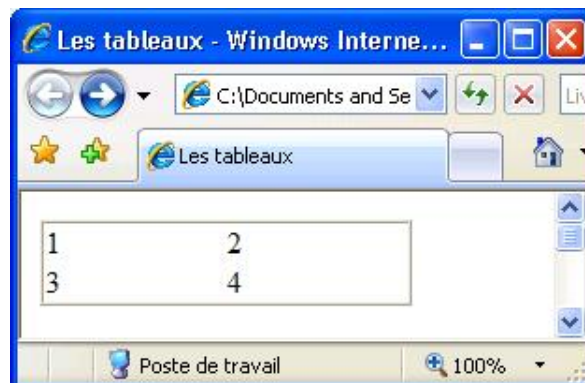
Résultat



Exemple d'un tableau avec uniquement les bords extérieurs et sans bords intérieurs :

```
<body>
<table border="1" frame="box" rules="none" width="75%" cellspacing="0">
<tr>
<td width="50%">1</td>
<td width="50%">2</td>
</tr>
<tr>
<td width="50%">3</td>
<td width="50%">4</td>
</tr>
</table>
</body>
```

Résultat



Commentaires

- Les feuilles de style abordées au chapitre Les feuilles de style apportent également de nouvelles présentations de bordures.

- Les effets les plus spectaculaires sont obtenus en combinant les attributs frame et rules.
- Ces attributs frame et rules sont implantés dans Internet Explorer et dans Firefox.

Diviser un tableau en groupes de colonnes

Pour mettre en forme (par exemple en gras et en couleur) toutes les cellules d'une colonne, le Html 3.2 n'offrait d'autre alternative que de mettre en forme chaque cellule, une par une ; ce qui représentait un travail long et rébarbatif.

Avec la balise `<colgroupspan="x">` où x est le nombre de colonnes visées, il est possible de grouper des cellules pour appliquer simultanément une même mise en forme à l'ensemble des cellules du groupe. Cette mise en forme peut se faire directement par des attributs ou par une feuille de style (cf. chapitre Les feuilles de style).

Cette balise `<colgroup>` se positionne juste après la balise `<table>` et avant toutes les autres balises `<tr>` et `<td>`.

Exemple

```
<body>
<table width="150" border="1">
<colgroup span=2 align="center" bgcolor="#FF0000">
<tr>
<td width="33%">1</td>
<td width="33%">2</td>
<td width="33%">3</td>
</tr>
<tr>
<td width="33%">4</td>
<td width="33%">5</td>
<td width="33%">6</td>
</tr>
</table>
</body>
```

Résultat



Commentaires

- Dans ce tableau, un groupe de deux colonnes a été créé pour déterminer **en une seule fois**, un alignement centré de chaque cellule et un fond rouge.
- Une fois n'est pas coutume, c'est Internet Explorer qui interprète le mieux cette balise. Firefox "oublie" les mises en forme éditées en Html mais prend bien en compte celles spécifiées en feuilles de style CSS.

Diviser un tableau en groupes de lignes

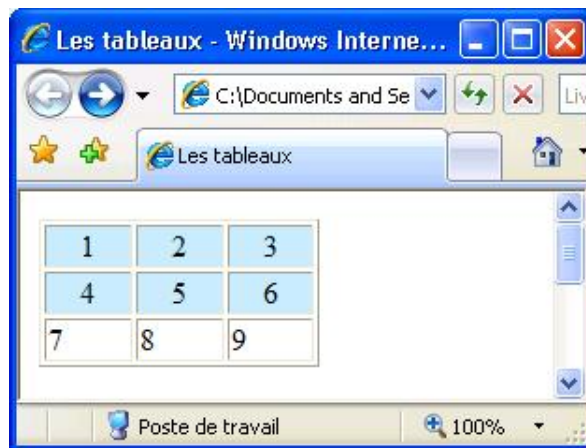
Il est également possible de créer une section horizontale d'une ou plusieurs lignes dans un tableau et de la mettre en forme en une seule opération.

Avec la balise `<tbody> ... </tbody>`, il est possible de grouper des lignes pour appliquer simultanément une même mise en forme à l'ensemble des cellules de la section. Cette mise en forme se réalise par des attributs de la balise ou par une feuille de style (cf. Chapitre Les feuilles de style). La balise de fermeture `</tbody>` indique la fin de section.

Exemple

```
<body>
<table width="150" border="1">
<tbody align="center" bgcolor="#FF0000">
<tr>
<td width="33%">1</td>
<td width="33%">2</td>
<td width="33%">3</td>
</tr>
<tr>
<td width="33%">4</td>
<td width="33%">5</td>
<td width="33%">6</td>
</tr>
</tbody>
<tr>
<td width="33%">7</td>
<td width="33%">8</td>
<td width="33%">9</td>
</tr>
</table>
</body>
```

Résultat



Commentaires

- Dans ce tableau, une section de deux lignes a été créée pour déterminer **en une seule fois**, un alignement centré et un fond de couleur.
- Les balises de sections horizontales se placent après les balises de groupes de colonnes.
- La balise `<tbody>` est bien reconnue par Internet Explorer et par Firefox.

Introduction

Le langage Html génère des documents mais sa véritable richesse se trouve dans sa fonction hypertexte qui permet de relier les pages Html pour en faire un site Web ou pour les mettre en relation avec d'autres pages du World Wide Web et avec les différents services d'Internet.


La balise de liens

La balise de liens est d'une simplicité étonnante.

```
<a href="destination">texte du lien</a>
```

Pour cette balise de liens, toute son efficacité réside dans la façon d'encoder la destination de sorte que le lien pointe :

- vers une autre page du site.
- vers un endroit de la page en cours.
- vers un endroit d'une autre page.
- vers une autre page située sur le Web.
- vers une adresse de courrier électronique.
- vers un fichier que le visiteur pourra télécharger.

 Cette balise de liens est également utilisée pour effectuer un lien à partir d'une image ou pour lancer un fichier son ou vidéo (cf. Chapitre Les images et arrière-plans - L'insertion d'un lien sur une image) ou pour lancer un fichier son ou vidéo (cf. Chapitre Le HTML et le multimédia).

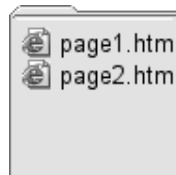
Les liens vers une autre page

1. Les liens vers une page située dans le même dossier

Lors de la création d'un site Web, le premier et meilleur conseil à donner est de regrouper **tous** les fichiers relatifs à son site dans le **même** dossier (répertoire). Ainsi, l'élaboration du lien sera la plus simple mais aussi la plus sûre.

Exemple

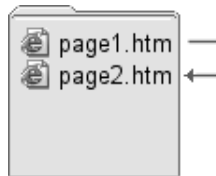
- Soient deux fichiers (page1.htm et page2.htm) situés dans le même dossier.



- Effectuez dans la page 1, un lien vers la page 2.

Le code du lien de la page 1 devient :

```
<a href="page2.htm">Vers page 2</a>
```



Commentaire

- Il est impératif de respecter scrupuleusement les majuscules et minuscules dans le nom du fichier et de son extension, car ce qui est reconnu par Windows ne l'est peut-être pas (ou sûrement pas) sous un serveur Unix.

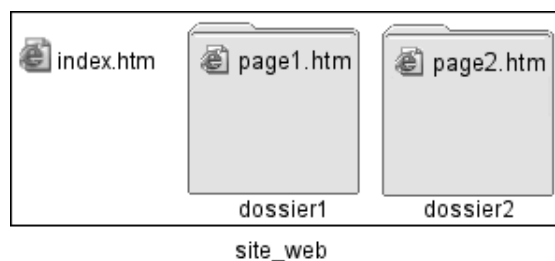
2. Les liens vers une page située dans un autre dossier

Dans les sites d'une certaine importance, on ne peut se contenter de mettre tous les fichiers dans un même et unique dossier. Pour structurer le site, il n'est pas rare d'avoir, dans un même dossier de départ, des sous-dossiers.

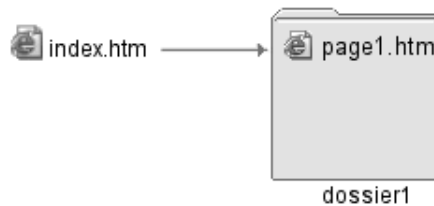
La bonne conception du chemin d'accès au fichier peut se révéler assez complexe, d'autant que la méthode d'adressage adoptée par les concepteurs est celle d'Unix, peu familière aux concepteurs habitués à un autre système d'exploitation et à Windows en particulier.

Exemple

Soit un dossier "site_web". Ce dossier comprend à la racine un fichier index.htm, un sous-dossier dossier1 avec le fichier page1.htm et un autre sous-dossier dossier2 avec le fichier page2.htm. Ce schéma nous permettra d'aborder plusieurs cas de figure.



- Effectuez d'abord un lien à partir de index.htm vers page1.htm situé dans le sous-dossier dossier1.



Le code du lien devient :

```
<a href="dossier1/page1.htm">Lien vers page 1</a>
```

Commentaires

- Index.htm et dossier1 se situent au même niveau de l'arborescence. Une fois dans dossier1, il suffit de "descendre" vers page1.htm.
 - Remarquez bien l'usage de la barre oblique / (*slash*) et non de la barre oblique inverse \ (*backslash*). Les anciens du MS-DOS, qui sont visés ici, se reconnaîtront.
- Effectuez maintenant un lien de page1.htm situé dans dossier1 vers page2.htm dans dossier2.



Le code du lien devient :

```
<a href="../dossier2/page2.htm">Lien vers page 2</a>
```

Commentaire

- Pour aller de page1.htm vers page2.htm, il faut d'abord remonter d'un niveau dans l'arborescence pour sortir de dossier1. Pour remonter d'un niveau, on utilise les signes "../". Une fois remonté d'un niveau, il faut descendre vers dossier2 et dans dossier2, aller chercher le fichier page2.htm.
- Imaginez un retour à partir de page2.htm vers le fichier index.htm.



Le code devient alors :

```
<a href="../../index.htm">Retour vers index</a>
```

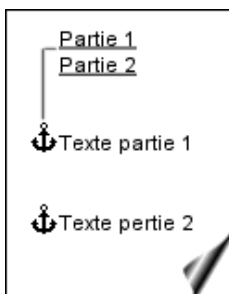
Commentaires

- En effet, pour aller de page2.htm vers index, il faut remonter d'un cran dans l'arborescence pour sortir du dossier dossier2 d'où ../. En redescendant, il est aisé de retenir index.htm.

- Les éditeurs Html vous simplifient grandement ce travail qui, dans le cadre d'une structure de site compliquée, relève vite du cauchemar.
- On parle d'adressage relatif pour les liens pointant vers une page située dans le même site.

Les liens à l'intérieur d'une page

Il est parfois intéressant de pouvoir mener l'utilisateur vers un endroit précis du document. On utilise alors ce qu'on appelle les ancres.



1. Les ancres

La syntaxe de création d'une ancre est :

```
<a name="nom de l'ancre"></a> ou  
<a name="nom de l'ancre">élément de la page</a>
```

Exemple

Insérez une ancre sur Texte partie 1

Le code devient alors :

```
<a name="ancre"></a>Texte partie 1
```

Commentaires

- Les ancres vont déterminer un repère de destination pour le lien.
- Outre le terme d'ancre, on rencontre aussi les termes de cible ou signet.

2. Les liens vers une ancre située dans la même page

Après avoir créé une ancre, vous pouvez définir un lien vers celle-ci.

Pour créer ce lien, il suffit de reprendre la balise de lien en précisant comme destination l'ancre souhaitée par : #nom de l'ancre.

Le code devient alors :

```
<a href="#ancre">Partiel</a>
```

3. Les liens vers une ancre située dans une autre page

Après avoir créé l'ancre, il faut définir l'adressage de la page (comme vu précédemment) et préciser le nom de l'ancre précédé du signe #.

Exemple

À partir du fichier *index.htm*, effectuez un lien vers un endroit précis du fichier *page1.htm* du dossier *dossier1*, défini par l'ancre ``.

Le code devient :

```
<a href="dossier1/page1.htm#ancre">Partiel</a>
```

Commentaire

- Il est utile de préciser qu'il n'y a pas d'espace entre page1.htm et le symbole #.

Les liens vers un autre site Web

Un lien peut faire référence à des pages d'autres sites, situées à une autre adresse sur le Web : la destination dans la balise de lien sera alors l'adresse complète du site ou de la page.

Exemple

```
<a href="http://www.lehtml.com/html/fichier.htm">Lien</a>
```

Commentaires

- Il s'agit bien de l'adresse complète, avec le protocole http://. En effet, avec les navigateurs récents, cette mention est devenue facultative et est ignorée par la plupart des utilisateurs.
- On parlera d'adressage absolu pour les liens pointant vers une page située dans un autre site.

Les liens vers une adresse électronique

Afin d'ajouter une touche d'interactivité à votre site et de permettre à vos visiteurs de vous contacter par courrier électronique, la destination dans la balise de lien sera alors l'adresse électronique, précédée du protocole du mail, soit mailto: (avec un double-point mais sans //).

```
<a href="mailto:editions@eni.com">L'auteur</a>
```

L'activation du lien, par le visiteur, ouvrira une fenêtre de l'application de sa messagerie électronique par défaut, par exemple Microsoft Outlook.

Bien qu'il ne s'agisse pas de code Html à proprement parler, il est aussi possible de prédéfinir un objet du message ainsi envoyé ou de prévoir l'envoi d'une copie à un autre destinataire.

Pour prédéfinir un objet à l'e-mail, le code devient :

```
<a href="mailto:editions@eni.com?subject=Formation Html">
Editions ENI</a>
```

où le contenu de subject est l'objet (prédéfini). Dans le cas de notre exemple, il s'agit de "Formation Html".

Pour définir une copie du message, le code devient :

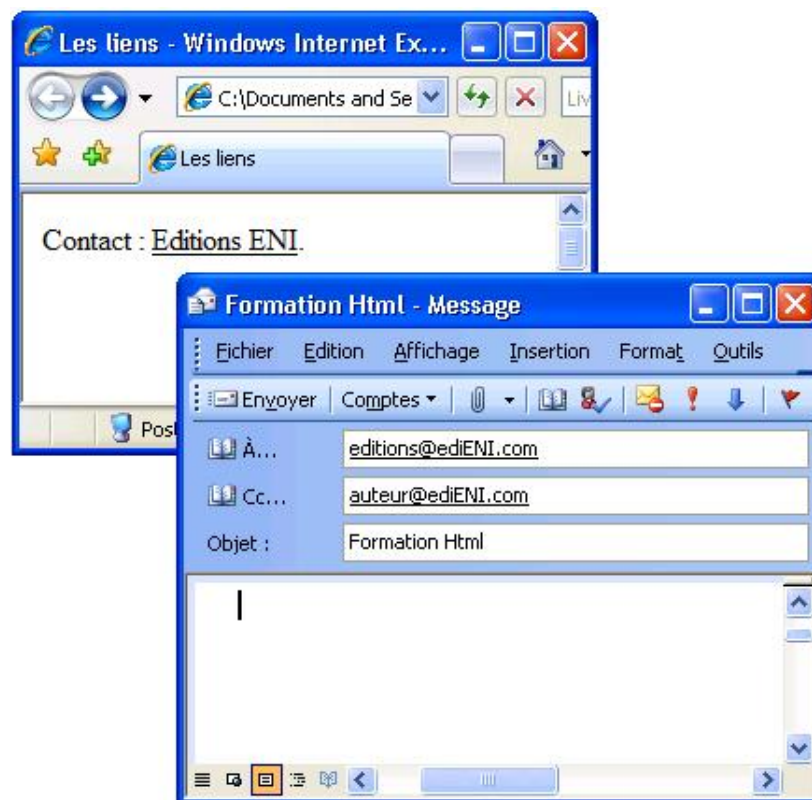
```
<a href="mailto:editions@eni.com?cc=auteur@eni.com">Editions ENI</a>
```

où le contenu de **Cc** est l'adresse électronique destinée à la copie.

Pour combiner les possibilités, le code devient :

```
<a href="mailto:editions@eni.com?subject=Formation Html&cc=auteur
@eni.com">Editions ENI</a>.
```

Résultat



Les liens pour télécharger un fichier

Vous souhaitez offrir à vos visiteurs la possibilité de télécharger un fichier pdf, un fichier compressé, un fichier son, un fichier vidéo, etc.

Il suffit pour cela, pour autant que le fichier téléchargeable soit **situé dans le même dossier**, de définir l'adresse du fichier comme destination dans la balise de lien.

Pour un fichier "formation.pdf" situé dans le même dossier, le code devient :

```
<a href="formation.pdf">Version pdf</a>
```

Si l'application Adobe Acrobat Reader qui lit les fichiers pdf n'est pas installée sur l'ordinateur du visiteur, le navigateur de celui-ci présentera la fenêtre d'invite de téléchargement. Par contre, si l'application Acrobat Reader est installée sur le poste de celui-ci, le browser ouvre l'application et affiche le fichier dans celle-ci.

Il en sera de même pour les autres types de fichiers. Quand il n'y a pas d'application définie par défaut pour l'extension du fichier, le navigateur le télécharge. Si une application est prévue par défaut pour ce genre de fichier, l'application est lancée automatiquement.

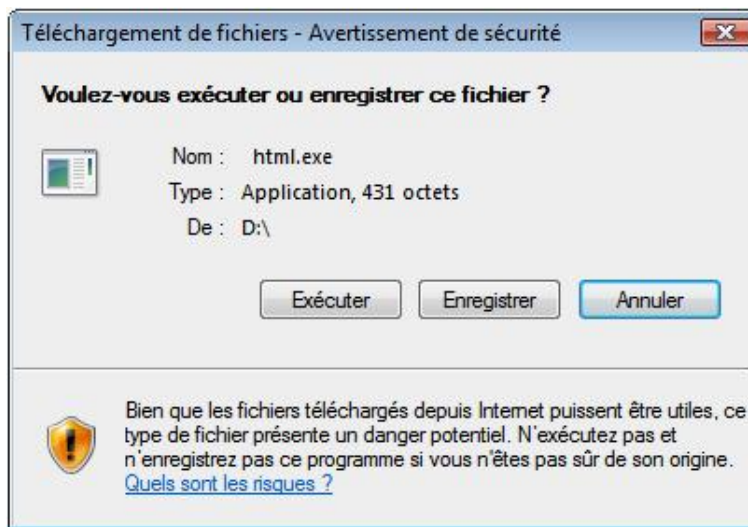
Dans le cadre du téléchargement, il est bien entendu possible de passer par le protocole ftp, pour autant que cette possibilité soit prévue par l'hébergeur de votre site.

Exemple

Pour télécharger un fichier exemple.zip, situé dans le même dossier que la page, le code est :

```
<html>
<head>
<title>Ma page</title>
</head>
<body>
<a href="html.exe">Version téléchargeable</a>
</body>
</html>
```

Résultat



Les liens vers une fenêtre spécifique (target)

Les pages ciblées par un lien peuvent s'ouvrir dans des fenêtres spécifiques grâce à l'attribut target.

Il offre les possibilités suivantes :

target="_self" (défaut)

La page cible du lien s'ouvre dans la même fenêtre ou sous-fenêtre que la page de départ du lien.

target="_top"

La page cible du lien est affichée dans la même fenêtre mais occupera la totalité de la fenêtre d'affichage (à ce stade de notre étude, self et top sont équivalents).

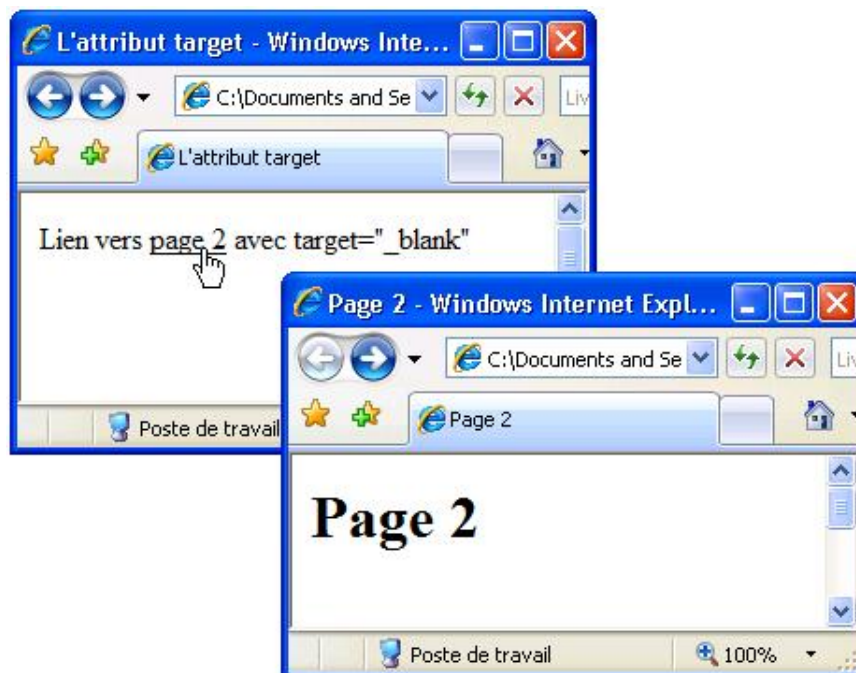
target="_blank"

La page cible du lien s'ouvre dans une nouvelle instance du navigateur soit dans une nouvelle fenêtre. Cet attribut est à utiliser avec circonspection car certains internautes ne pensent pas à revenir à l'instance précédente du navigateur.

Exemple dans le fichier Page 1

```
<html>
<head>
<title>Page 1</title>
</head>
<body>
Vers <a href="page2.htm" target="_blank">Page2</a>...
</body>
</html>
```

Résultat



➤ L'attribut target est surtout utilisé lors de la découpe de la fenêtre en différents cadres (nous y reviendrons plus longuement au chapitre Les cadres).

L'attribut title

L'attribut `title="Texte"` de la balise `<a>` permet de proposer une info-bulle explicative au lien.

Exemple

```
<a href="page2.htm" title="Lien vers page 2">Lien</a>
```

Résultat



La couleur des liens non-visités, visités et actifs

La couleur des liens varie suivant si le lien a déjà été visité ou non. Sauf spécifications de la part du concepteur, les liens non-visités s'affichent en bleu, les liens visités en mauve et les liens actifs au moment du clic par l'utilisateur en rouge.

Comme à l'origine de la toile, les pages Web avaient par défaut une sinistre couleur d'arrière-plan grise, ces couleurs avaient au moins l'avantage d'être visibles. Pour peu que vous utilisiez un fond de page de couleur foncée ou que vous souhaitiez apporter une touche d'originalité, il devient indispensable de modifier ces couleurs de lien par défaut.

Bien qu'ils affectent les liens, les attributs pour modifier la couleur des liens sont des attributs de la balise `<body>`.

En Html, la couleur des liens est déterminée de façon uniforme pour l'ensemble du document.

Les attributs pour changer la couleur sont :

`link="#$$$$$"` pour le lien non-visité.

`vlink="#$$$$$"` pour le lien déjà visité.

`alink="#$$$$$"` pour le lien actif.

Dans cette même balise `<body>`, on peut aussi déterminer la couleur par défaut de la page par l'attribut :

`text="#$$$$$"` pour la couleur du texte.

Par exemple :

```
<body link="#00FF00" vlink="#FFFF00" alink="#FF00FF">
```

-
- Il est cependant conseillé de ne pas modifier trop souvent la couleur des liens d'une page à l'autre afin d'assurer une cohérence graphique et logique à votre site.
-

Les liens et les feuilles de style

Les feuilles de style (cf. Chapitre Les feuilles de style) vont apporter un peu de fantaisie dans la présentation des liens. Citons les possibilités :

- supprimer le soulignement par défaut.
- définir des couleurs de liens différentes dans un même document.
- mettre le texte du lien en gras ou en italique lors du survol du lien avec la souris.
- modifier la taille, la police ou la couleur lors du survol du lien avec la souris.
- ajouter un arrière-plan au lien toujours lors du survol de la souris.
- prévoir des couleurs de liens différentes sur la même page.
- etc.

Les couleurs en Html

1. La cotation en hexadécimal

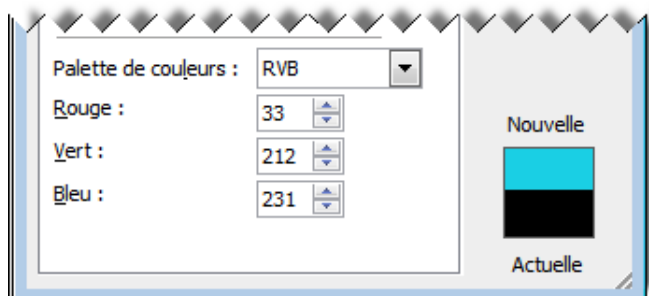
Comme nous l'avons déjà évoqué dans le chapitre Le texte et sa présentation, les couleurs sont le plus souvent notées selon le système RVB (Rouge, Vert, Bleu), avec pour chaque composante un chiffre décimal de 0 à 255. Le Html a préféré exprimer ces composantes en valeur hexadécimale (base 16) soit de 00 à FF. Cette notation plus compacte présente l'avantage d'exprimer en 6 positions l'ensemble des couleurs disponibles. Cette notation en hexadécimal apparaît être, au premier abord, assez déconcertante.

Mais comment trouver cette valeur hexadécimale ?

a. La conversion par la calculatrice

Pas très élaborée sur le plan informatique, cette solution possède au moins l'avantage de pouvoir calculer la valeur hexadécimale sans applications graphiques.

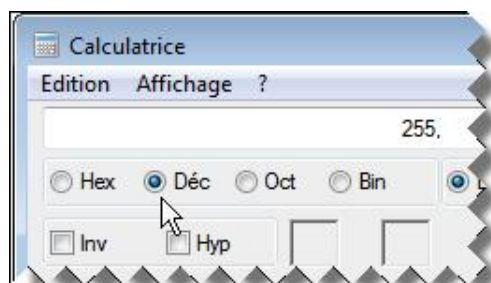
Les applications de Windows qui proposent l'écran de couleur avec les valeurs RVB sont nombreuses (par exemple dans Microsoft Word).



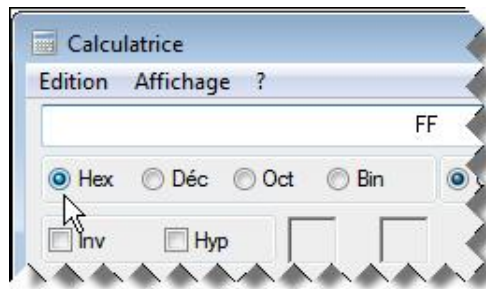
- Convertissez chaque valeur en hexadécimal par l'application "Calculatrice" de Windows.
- Dans le menu **Démarrer - Programmes - Accessoires - Calculatrice**, activez l'option **Scientifique**.



- Encodez la valeur à convertir en cliquant sur les touches de chiffres avec le bouton **Dec** actif (par défaut).



- Pour obtenir la valeur hexadécimale, cliquez sur le bouton **Hex**.



En Annexe, un tableau de conversion des valeurs décimales de 0 à 255 en valeur hexadécimale.

b. Des petites applications de conversion

Il est possible de télécharger de petits programmes qui assurent la conversion de la couleur en hexadécimal ; ainsi, Hexacolor, gratuit pour les particuliers, que vous trouvez sur le site www.htmledit.com et Color Tohexa par exemple, entièrement freeware.



Un double clic sur le rectangle de couleur dans **Couleur** permet de changer de couleur.

c. Les applications graphiques

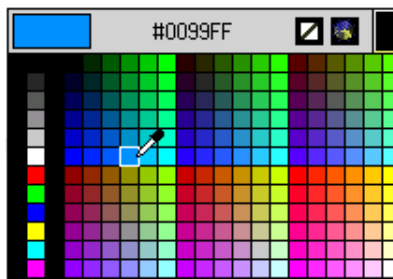
Toutes les applications graphiques comme Adobe Photoshop, Paint Shop Pro, Macromedia Fireworks... proposent la valeur RVB des couleurs et son code hexadécimal.

d. Les éditeurs Html

Tous les éditeurs Html proposent bien entendu la possibilité de choisir n'importe quelle couleur avec son code hexadécimal.

Certains éditeurs Html comme FrontPage proposent même directement quelques outils graphiques, limités mais suffisants dans le cadre de la création de site, comme la transparence, l'ajout de texte sur l'image, la conversion de format d'image, la sélection d'une partie de l'image, etc.

À titre d'exemple, voici la fenêtre de couleur de Adobe Dreamweaver CS4.



e. Des sites Web

De nombreux sites destinés aux webmasters proposent des pages qui détaillent l'éventail des couleurs et leur correspondance en hexadécimal.

Quelques pages Web consacrées à l'illustration des couleurs se trouvent dans les exemples de cet ouvrage, téléchargeables à l'adresse www.editions-eni.com/exemples dans le dossier "couleurs".

➤ Les feuilles de style (cf. Chapitre Les feuilles de style) offrent, parmi d'autres possibilités de notation des couleurs, une notation en RGB nettement plus conviviale. Par exemple le rouge pourra être noté ; color : rgb(255,0,0).

2. La désignation par un nom

Pour désigner des couleurs, leur nom peut être utilisé. Ces noms de couleurs sont en anglais et il faut déjà une connaissance pointue de l'anglais pour percevoir toutes les nuances entre des noms de couleur comme crimson, gainsboro, oldlace, etc.

En outre, ces noms n'ont pas été normalisés par le W3C et chaque navigateur (Internet Explorer ou Firefox) a sa propre liste de nom et ne reconnaît pas les noms de couleur du concurrent. Les risques d'incompatibilité sont donc très élevés.

Les professionnels de la conception de sites Web restent d'ailleurs fidèles à la codification en hexadécimal sauf peut-être pour quelques grands classiques.

Couleur	hexadécimal	nom
Noir	#000000	black
Marron	#800000	maroon
Vert bouteille	#008000	green
Vert Olive	#808000	olive
Bleu marine	#000080	navy
Violet	#800080	purple
Cyan foncé	#008080	teal
Gris clair	#C0C0C0	silver
Gris moyen	#808080	gray
Rouge	#FF0000	red
Vert clair	#00FF00	lime
Jaune	#FFFF00	yellow

Bleu	#0000FF	blue
Fuchsia	#FF00FF	fuchsia
Bleu clair	#00FFFF	aqua
blanc	#FFFFFF	white

➤ Les feuilles de style (cf. Chapitre Les feuilles de style) offrent, parmi d'autres possibilités de notation des couleurs, une notation en RGB nettement plus conviviale. Par exemple le rouge pourra être noté ; color : rgb (255,0,0).

Les images du Web

Les images ont grandement contribué à la popularité de Web, créant ainsi son aspect convivial et attractif qui en fait un univers de création et de communication à part entière.

Pour réduire au maximum le temps de transmission, le Web réclame un format d'image réduit et donc compressé.

Avant d'inclure vos images, il est souvent indispensable de les convertir dans un format (compressé) compatible avec le Web, d'en optimiser la dimension et surtout la taille en octets. Cette modification ne peut se réaliser que par des programmes graphiques comme Adobe Photoshop, Paint Shop Pro ou Macromedia Fireworks ou Gimp.

En ce qui concerne la conception de pages Web, le plus grand réservoir d'outils se trouve justement sur le Web. Ainsi, on peut facilement rapatrier des images ou "cliparts" en pointant le curseur de la souris sur l'image et en cliquant sur l'option **Enregistrer** du menu contextuel qui se présente alors.



Une recherche avec le mot clé "clipart" devrait vous mener à des sites où des collections d'images sont mises gracieusement à votre disposition. Ces images sont déjà au bon format.

Attention toutefois, il peut exister un copyright sur certaines images et la courtoisie veut que l'on demande la permission de les utiliser pour une diffusion sur la toile.

Le Web ne supporte que trois formats d'image soit le GIF, le JPEG et le PNG.

1. Le format GIF

Le format Gif pour *Graphics Interchange Format* a été développé par CompuServe en 1987 (soit avant l'invention du Html) pour permettre un échange rapide de fichiers graphiques sur les lignes téléphoniques. Il fit très vite le bonheur de l'Internet et a été tout naturellement repris par le langage Html du Web.

Les principales caractéristiques du format Gif sont :

- C'est un format de compression efficace et rapide.
- La conversion en Gif n'entraîne pas de perte de qualité.
- Il permet de rendre une couleur transparente (GIF89a).
- Il permet aussi un effet d'affichage progressif, dit entrelacé.
- Il permet des animations, sortes de petits dessins animés, avec ce qu'on appelle les Gifs animés.



L'inconvénient majeur du Format Gif est qu'il ne permet de coder les images qu'en 256 couleurs ou moins.

2. Le format JPEG

Le format JPEG a été mis au point par le *Joint Photographic Expert Group* pour répondre au besoin de disposer d'un moyen de compression pour les images photographiques de haute qualité.

Les principales caractéristiques du format Jpeg sont :

- Il permet d'afficher jusqu'à 16,7 millions de couleurs.
- C'est un format "à la carte" car on peut en faire varier le taux de compression.
- Il permet aussi maintenant un affichage progressif.

Les inconvénients du Jpeg sont :

- La compression entraîne une perte de qualité.
- Il ne permet pas de revenir à la qualité initiale une fois la compression enregistrée.



Avec un taux de compression de 25% et une qualité de 75%, on se retrouve souvent au compromis idéal entre compression (donc taille en octets) et qualité.

3. Le format PNG

Le format PNG pour *Portable Network Graphic*, est un nouveau format qui a fait son apparition récemment. Son utilisation est encore peu répandue mais va cependant croissante. Le format PNG se caractérise par ses qualités de compression tout en gardant une excellente qualité. Le W3C le présente comme le format d'image de l'avenir.

4. Gif ou Jpeg ?

Bien qu'il n'y ait pas de règles absolues, voici quelques éléments qui peuvent influencer le choix entre Gif et Jpeg.

Le format Gif est idéal pour les images informatiques comme les icônes, bannières ou logos tandis que le Jpeg est quant à lui très bien adapté aux images photographiques ou scannées.

Le Gif se révèle très performant pour les petites images aux couleurs limitées et marquées alors que le Jpeg excelle pour les images fouillées avec des dégradés ou des nuances de couleurs proches.

Le Jpeg s'impose comme indispensable pour les grandes images et les photos.

L'insertion d'une image

La balise d'insertion d'image est ``.

Le fichier_image est le nom et l'extension de l'image, si elle est située dans le même dossier que le fichier Html. Dans le cas contraire, il faut en outre spécifier le chemin d'accès au fichier_image (comme étudié dans le chapitre Les liens). En cas d'erreur dans le nom du fichier et/ou dans l'adressage, l'image ne sera pas affichée.

Cette balise image comporte des attributs forts importants :

1. L'attribut alt

L'attribut `alt="texte de remplacement"` était à l'origine prévue pour fournir un texte de remplacement aux navigateurs non graphiques ou aux navigateurs dont l'option d'affichage des images a été désactivée. Depuis, ce texte a été utilisé par certains navigateurs (Internet Explorer) pour fournir une info-bulle explicative du meilleur effet lors du survol de l'image par la souris. Plus récemment encore, le contenu de l'attribut alt est utilisé dans la base de données des moteurs de recherche (Google) pour leur catégorie Image. Étant donné ces multiples utilités, pourquoi s'en priver ? La spécification Html 4.0 le considère même comme obligatoire. Il l'est également en XHTML (cf. Chapitre Le XHTML).

L'attribut alt est non seulement obligatoire mais également très utile dans le domaine de l'accessibilité des sites Web. Le contenu de l'attribut est, par exemple, lu par les synthèses vocales utilisées par les personnes non-voyantes et permet à celles-ci de prendre connaissance du contenu informatif des images.

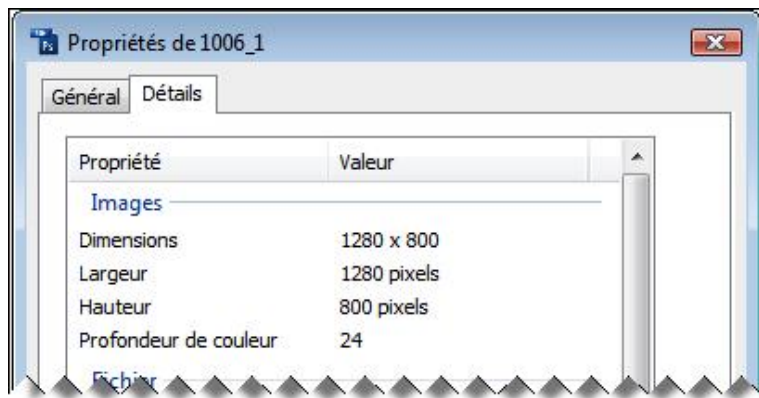
2. Les attributs width et height

Un navigateur n'a beau être qu'une visionneuse de page Web, sa tâche n'est pas pour autant facile. En effet, à partir du code source du document, il doit effectuer sa mise en page pour afficher du texte, des images, des tableaux, etc. En spécifiant la largeur et la hauteur de l'image par les attributs `width` et `height`, il pourra réserver son emplacement et continuer à composer et à afficher le texte avant son chargement complet. En définissant la dimension de l'image, le chargement de la page est accéléré et donne matière à patienter aux internautes. Bien que facultatifs, ces attributs sont essentiels pour tous les concepteurs professionnels.

La définition de la taille se fait par les attributs `width="x"` et `height="y"` où x est la largeur en pixels de l'image et y la hauteur en pixels.

On devra généralement avoir recours à une application graphique pour connaître la taille.

➤ Windows XP et Vista proposent directement les dimensions d'une image dans son menu contextuel. Effectuez un clic droit sur l'image, choisissez l'option **Propriétés** et activez l'onglet **Détails** pour Vista ou **Résumé** pour XP.



3. L'attribut border

On peut éventuellement ajouter un bord à l'image par l'attribut `border="x"` où x est exprimé en pixels.

Cet attribut n'est que rarement utilisé, car on ne peut en définir la couleur. Même si nous y revenions au point H (Insérer un lien sur une image), cet attribut est souvent défini à 0 soit `border="0"` pour enlever le bord introduit par défaut pour les images servant de lien.

4. L'attribut lowsrc

Initialement introduit par Netscape, cet attribut est maintenant aussi reconnu par Explorer et Firefox. Cet attribut permet de charger une première image qui est généralement d'une résolution plus basse et donc d'un téléchargement plus rapide (car moins gourmande en octets). C'est cette image qui est chargée en premier.

Le navigateur s'occupe ensuite du chargement de l'image définitive et l'affiche en lieu et place de la première. Les deux images seront bien entendues de la même dimension.

Cet attribut, assez marginal, n'appartient pas à la norme Html 4.0.

Exemple général de la balise

Soit une image *gsm.gif* qui se situe dans le même dossier que mon fichier *htm*. Les dimensions en sont 23 pour la largeur et 36 pour la hauteur. Je souhaite inclure cette image dans ma page.

Exemple

```
<html>
<head>
<title>Les images</title>
</head>
<body>
<p align="center">

</p>
</body>
</html>
```

Résultat



Commentaire

- La balise image est une de ces balises uniques du Html et ne comporte donc pas de balise de fermeture.
- En Html, les images sont des fichiers séparés par rapport à la page Web.

L'espace autour d'une image

Il est possible d'ajouter une zone (vide) autour d'une image pour éviter qu'elle ne soit collée au texte ou à tout autre élément de la page.

L'attribut de la balise image `hspace="x"` où `x` est exprimé en pixels, ajoute horizontalement de l'espace à gauche et à droite de l'image.

L'attribut de la balise `vspace="x"`, ajoute verticalement de l'espace en haut et en bas de l'image.

Exemple

```
<html>
<head>
<title>Les images</title>
</head>
<body>
<p align="center">
Mon PDA
</p>
<p align="center">
Mon PDA
</p>
</body>
</html>
```

Résultat



L'alignement d'une image

La balise image comporte également des attributs spécifiques pour l'alignement de celle-ci par rapport au texte ou aux autres éléments de la page.

L'alignement se réalise par l'attribut `align="type"` où `type` prend la valeur :

`top` Qui aligne l'image sur le plus haut élément de la ligne.

`texttop` Qui aligne l'image sur le plus haut du texte de la ligne.

`absmiddle` Qui centre l'image sur le milieu du plus grand élément.

`middle` Qui centre sur la ligne de base.

`absbottom` Qui aligne le bas de l'image sur le bas du plus grand élément.

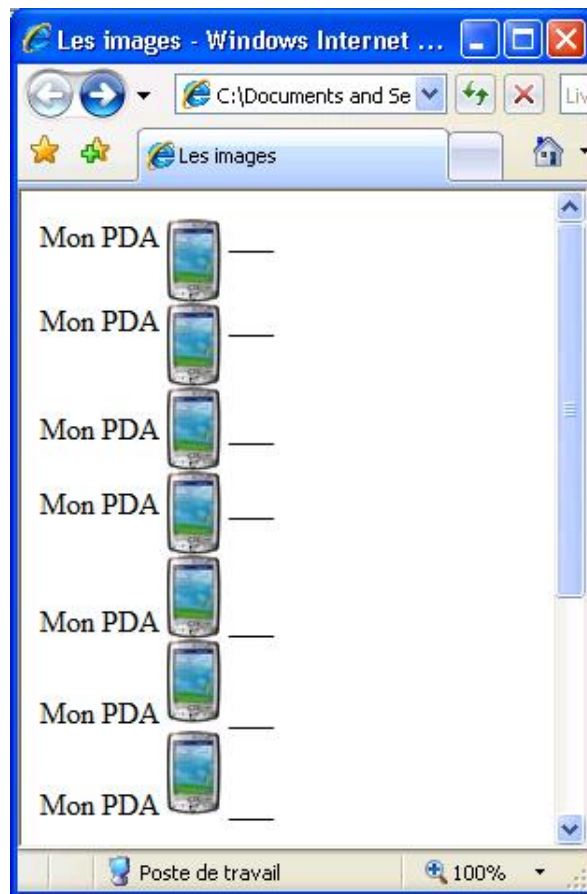
`bottom` Qui aligne le bas de l'image sur le plus bas du texte.

`baseline` Qui aligne le bas de l'image sur la ligne de base.

Exemple

```
<html>
<head>
<title>Les images</title>
</head>
<body>
Mon PDA  ____<br>
Mon PDA  ____<br>
Mon PDA  ____<br>
Mon PDA  ____<br>
Mon PDA  ____<br>
Mon PDA  ____<br>
Mon PDA  ____
</body>
</html>
```

Résultat



Commentaire

- Les navigateurs se révèlent parfois un peu "paresseux" lorsqu'il s'agit d'aligner plusieurs images avec un alignement différent sur la même ligne. Les résultats sont alors imprévisibles.

L'insertion d'une couleur d'arrière-plan

Pour donner une couleur d'arrière-plan (unie) à la page, l'attribut `bgcolor="#$$$$$"` ou `bgcolor="nom"` est ajouté à la balise `<body>`.

Exemple

```
<html>
<head>
<title>Les images</title>
</head>
<body bgcolor="#99CCFF">
</body>
</html>
```

Résultat



Commentaire

- En donnant une couleur d'arrière-plan à la page, il faut veiller à ce que la couleur des caractères du texte soit adaptée à la couleur de fond. Trop de pages possèdent un texte dont le coloris est trop proche de celui du fond de la page et empêche ainsi sa lisibilité. Cette remarque vaut aussi pour les images d'arrière-plan.

L'insertion d'image d'arrière-plan

1. Dans une page

Une image peut être utilisée pour tapisser le fond de la page en mosaïque.

Pour cela, à la balise <body> est ajouté l'attribut background="fichier_image" où fichier_image correspond au nom de votre image et éventuellement à l'adressage pour y accéder.

Bien entendu, il est préférable de mettre une petite image (effet de mosaïque). Mettre comme fond de page une photo panoramique qui occupe tout le fond d'écran est une hérésie. Attention au temps de chargement !

Ajoutez l'image d'arrière-plan gris.jpg, située ici dans le même dossier que le fichier htm.

Exemple

```
<html>
<head>
<title>Les images</title>
</head>
<body background="gris.jpg">
</body>
</html>
```

Résultat



Commentaire

- Votre fond de page ne doit jamais altérer la lisibilité du texte. Les feuilles de style apportent en la matière de larges possibilités (image fixe, filigrane).

2. Dans un tableau

Il est possible d'insérer une image d'arrière-plan à un tableau (cf. Chapitre Les tableaux). Il suffit d'ajouter l'attribut background="fichier_image" à la balise <table>.

Exemple

```
<html>
<head>
<title>Les images</title>
</head>
<body>
<table width="200" background="gris.jpg" border="1">
<tr align="center">
<td>1</td><td>2</td>
</tr>
<tr align="center">
<td>3</td><td>4</td>
```

```

</tr>
</table>
</body>
</html>

```

Résultat



Dans le même ordre d'idée, il est aussi possible d'ajouter une image d'arrière-plan à une cellule, en ajoutant l'attribut `background= "fichier_image"` à la balise `<td>`.

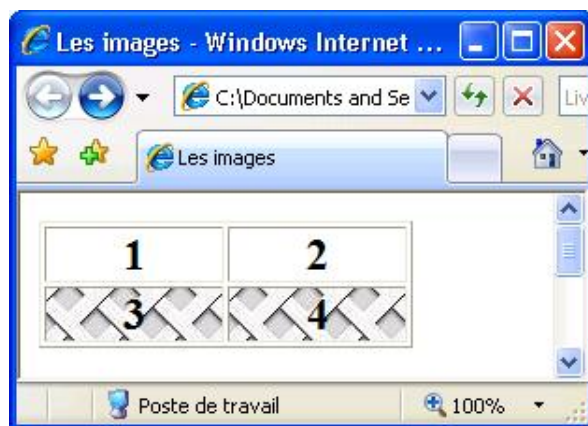
Exemple

```

<html>
<head>
<title>Les images</title>
</head>
<body>
<table width="200" border="1">
<tr align="center">
<td>1</td><td>2</td>
</tr>
<tr align="center">
<td background="gris.jpg">3</td>
<td background="gris.jpg">4</td>
</tr>
</table>
</body>
</html>

```

Résultat



Commentaire

- Il y a de légères différences d'interprétation entre Explorer et Netscape quant à l'affichage de l'image d'arrière-plan dans un tableau ou une cellule.

L'insertion d'un lien sur une image

Insérer un lien ne pose pas de problèmes particuliers. Il suffit d'encadrer la balise image avec les balises de lien.

Ce qui décontenance le programmeur débutant, c'est le cadre de couleur que le navigateur ajoute par défaut pour signaler le lien hypertexte.

Il suffit d'ajouter l'attribut `border="0"` à la balise `` pour faire disparaître le cadre.

Exemple

```
<html>
<head>
<title>Les images</title>
</head>
<body>
<p>
<a href="#"></a>
</p>
<p>
<a href="#"></a>
</p>
</body>
</html>
```

Résultat



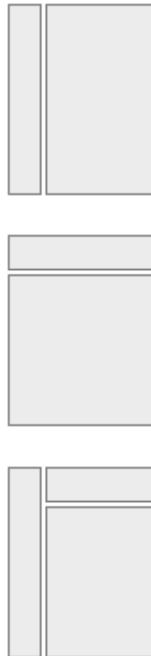
Les images réactives

Nous aurions pu parler dans ce chapitre des images réactives qui sont des images que l'on peut diviser en différentes zones auxquelles on peut associer un lien vers une page distincte.

Un chapitre entier est consacré à ces images réactives car le code Html à mettre en œuvre est très spécifique et surtout très éloigné des balises d'images "normales" abordées dans le présent chapitre (cf. Chapitre Images réactives).

Présentation

L'espace réservé par le navigateur peut être divisé en plusieurs zones distinctes qui permettent d'afficher simultanément plusieurs pages Html. Cette fonction est réalisée par les cadres, appelés aussi "frames". Cette façon de diviser l'écran peut à la limite, ne pas être perceptible par le visiteur.



Les cadres proposent aux concepteurs de sites une possibilité puissante et illimitée de mise en page. En outre, comme il est possible par le système de liens hypertextes d'influer à partir d'une page de cadres sur une autre page de cadres, la diffusion et la transmission de l'information peuvent ainsi prendre une véritable dimension hypertextuelle propre au Web.

Cette richesse dans la présentation et l'organisation de l'information se traduit par un code pointu, délicat à mettre en place et où toute erreur, spécialement dans l'élaboration des liens, se paiera comptant.

En outre, comme les cadres ne font officiellement partie du Html standard que depuis la version Html 4.0, ils peuvent éventuellement présenter des problèmes de compatibilité avec certains navigateurs qui ne prendraient pas en charge cette fonction. Mais en ce début du 21ème siècle, on peut raisonnablement supposer qu'ils sont rarissimes.

Définir une page de cadres élémentaire

La structure d'une page de cadres, appelée aussi jeu de cadres, est différente du document Html classique.

Exemple





```
<html>
<head>
</head>
<frameset>
-
Les balises propres aux cadres.
-
</frameset>
<noframes>
Les informations pour les navigateurs
qui ne reconnaissent pas les cadres.
</noframes>
</html>
```

Commentaires

- La balise `<frameset>` remplace la balise `<body>` du document Html classique. Elle indique au navigateur que la disposition s'effectue en plusieurs sous-fenêtres. La balise `<frameset>` est bien entendue complétée par de nombreux attributs détaillés dès le point suivant.
- Le contenu des balises `<noframes>` ne sera affiché que par les navigateurs qui ne reconnaissent pas les cadres. En absence de ce contenu, ces navigateurs affichent une page vide et sans message d'erreur.
- Bien que les cadres ne fassent officiellement partie du Html que depuis le Html 4.0, ils étaient cependant déjà reconnus depuis Netscape 2 et Explorer 3. La balise `<noframes>` peut alors sembler d'une utilité limitée... Cependant, elle reste d'actualité vis-à-vis des moteurs de recherche dont certains utilisent le contenu pour le référencement de votre site (cf. Les cadres et le référencement par Google de ce chapitre).

Ainsi, il est prudent de prévoir d'autres informations que le sempiternel : "Ce site contient des cadres".

➤ Il est conseillé pour la suite du travail de préparer, **dans un même répertoire**, les différents fichiers utilisés dans les exemples : la page de jeu de cadres appelée `cadre.htm` et les fichiers à insérer dans les cadres soit `page1`, `page2` et `page3.htm`, etc.

 cadre.htm
 page1.htm
 page2.htm
 page3.htm

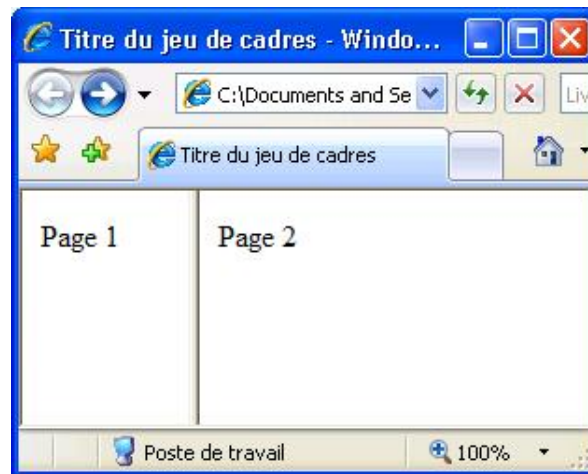
Créer un cadre avec des colonnes

Créez un jeu de cadres avec deux colonnes où la colonne de gauche contient le fichier page1.htm et la colonne de droite, le fichier page2.htm.

Exemple

```
<html>
<head>
<title>Titre du jeu de cadres</title>
</head>
<frameset cols="30%, 70%">
<frame name="gauche" src="page1.htm">
<frame name="droite" src="page2.htm">
</frameset>
</html>
```

Résultat



Commentaires

- Le titre du jeu de cadres est celui de la page de cadre et de toutes les fenêtres qui utiliseront ce jeu de cadres. Il prend donc toute son importance informative pour le visiteur.
- À la balise `frameset`, l'attribut `cols` est ajouté (pour colonnes) pour diviser la fenêtre en colonnes.
- Il faut encore indiquer au navigateur le nombre et la dimension de ces colonnes : `cols="largeur1,largeur2,largeur3,..."`
 - Les données pour la division en colonnes doivent être mises entre guillemets et séparées par des virgules.
 - La dimension peut être exprimée en pourcentage, ici `cols="30%, 70%"`. Ce qui est assez pratique car le Webdesigner ignore la résolution d'écran employée par le visiteur final et la division ainsi choisie s'adaptera toujours à cette dernière. Il est important d'avoir un total de 100% pour éviter toute correction inappropriée du navigateur.
 - La dimension peut être exprimée en pixels ; par exemple `cols="200,600"` qui s'adapte à merveille à une résolution d'écran de 800x600 mais dont le résultat risque d'être aléatoire avec une résolution d'écran de 1024x768.
 - Vous avez aussi droit à un joker en utilisant le signe étoile * pour déterminer la dimension d'une fenêtre. Soit par exemple `cols="200,*"`. Ainsi par cette notation dite dynamique, la colonne de gauche aura toujours une largeur de 200 pixels tandis que la colonne de droite prendra le reste de la fenêtre du navigateur et ce, quelle que soit la résolution d'écran de votre visiteur. Dans le cas de plusieurs colonnes, il est possible d'utiliser plusieurs signes *. Dans ce cas, le navigateur affiche pour chaque signe * des colonnes de largeur identique.

- La balise `<frame>` se rapporte au contenu des divisions de la fenêtre principale commandées par la balise `frameset`.
 - Il est préférable de donner un nom à chacune des sous-fenêtres ainsi déterminées par l'attribut `name="nom"`. Ce nom associé à l'attribut `target` sera bien utile pour désigner la fenêtre cible des liens (cf. section Créer des liens entre les cadres, dans ce même chapitre).
 - L'attribut `src="page.htm"` est l'adresse de la page qui s'affichera dans le cadre.



Il existe d'autres attributs possibles, détaillés tout au long de ce chapitre.

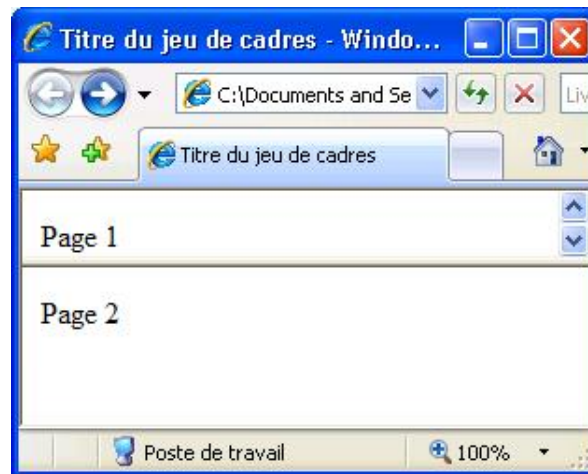
Créer un cadre avec des lignes

Créez un jeu de cadres avec deux lignes où la ligne supérieure contiendra le fichier page1.htm et la colonne inférieure le fichier page2.htm.

Exemple

```
<html>
<head>
<title>Titre du jeu de cadres</title>
</head>
<frameset rows="30%,70%">
<frame name="sup" src="page1.htm">
<frame name="inf" src="page2.htm">
</frameset>
</html>
```

Résultat



Commentaires

- À la balise `frameset` est ajouté l'attribut `rows` (pour rangées) pour diviser la fenêtre en lignes. Soit `rows="hauteur1,hauteur2,hauteur3..."`.
- Tout ce qui a été dit pour les colonnes (`cols`) est bien entendu valable pour les lignes (`rows`).
- Le navigateur ajoute automatiquement par défaut une barre de défilement lorsque le cadre est trop petit pour afficher la totalité de la page qu'il contient.

Créer un cadre avec des colonnes et des lignes

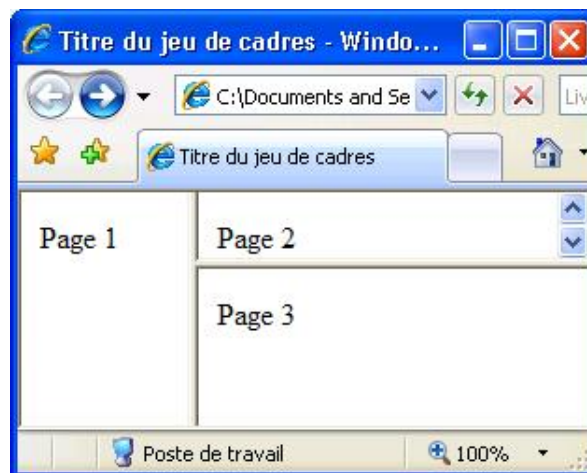
Créez un jeu de cadres avec des lignes et des colonnes. Pour cela, il faut imbriquer des balises `<frameset>`.

À l'écran, on perçoit que celui-ci est d'abord divisé en 2 colonnes (cols) et que la colonne de droite est ensuite divisée en deux lignes (rows).

Exemple

```
<html>
<head>
<title>Titre du jeu de cadres</title>
</head>
<frameset cols="30%,70%">
<frame name="gauche" src="page1.htm">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
</html>
```

Résultat



Commentaires

- Il est important de ne pas oublier la ou les balises `</frameset>` de fermeture car en cas d'oubli, seule une page blanche sera affichée.
- Toutes les combinaisons de lignes et colonnes sont possibles mais au risque d'augmenter la complexité du code.

Définir les attributs des cadres

1. Les bordures

Avec l'attribut `frameborder`, il est possible d'afficher ou de masquer les bordures.

La syntaxe en est simple :

`frameborder="0"` ou `"no"`

pour masquer les bordures.

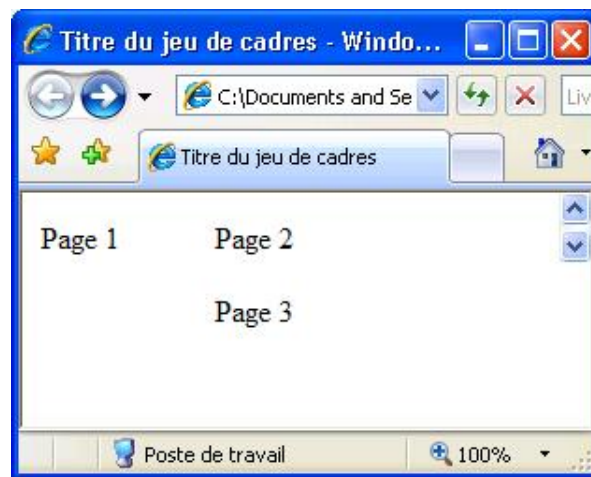
`frameborder="1"` ou `"yes"`

pour afficher les bordures (par défaut).

Exemple

```
<frameset cols="30%,70%" frameborder="0">
<frame name="gauche" src="page1.htm">
  <frameset rows="30%,70%" frameborder="0">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



Commentaire

- L'attribut `frameborder` peut également être affecté à la balise `<frame>`. L'attribut `frameborder=0` dans la balise `<frameset rows="30%,70%">` est inutile car la propriété d'absence de bordures est induite par le premier `frameset` affectant les colonnes. Certains pourraient être tentés de prévoir une bordure dans un cadre particulier. Le résultat risque cependant d'être difficile à mettre en œuvre car chaque cadre partage souvent sa bordure avec d'autres cadres adjacents.

2. La couleur des bordures

Avec l'attribut `bordercolor`, il est possible de définir la couleur des bordures.

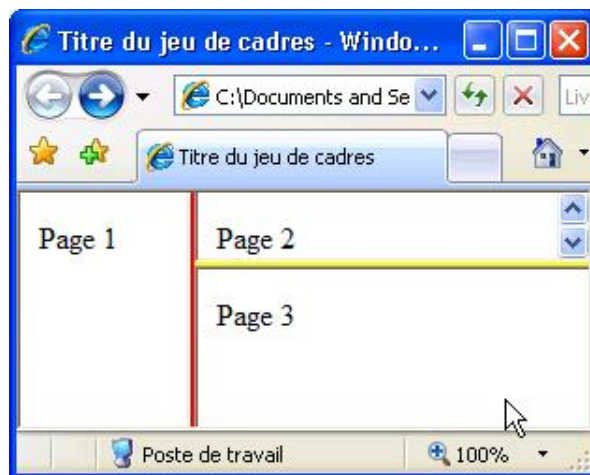
La syntaxe est `bordercolor="#$$$$$$"` ou `bordercolor="nom"` soit la couleur en valeur hexadécimale ou par nom.

Exemple

```
<frameset cols="30%,70%" bordercolor=red>
<frame name="gauche" src="page1.htm">
```

```
<frameset rows="30%,70%" bordercolor="#FFFF00">
<frame name="droite_sup" src="page2.htm">
<frame name="droite_inf" src="page3.htm">
</frameset>
</frameset>
```

Résultat



La bordure verticale est rouge tandis que la bordure horizontale est jaune.

Commentaires

- L'attribut `bordercolor` peut également être affecté à la balise `<frame>`.
- L'attribut `bordercolor`, bien que parfaitement reconnu par Internet Explorer et Firefox, ne fait pas partie de la norme Html 4.0.

3. L'épaisseur des bordures

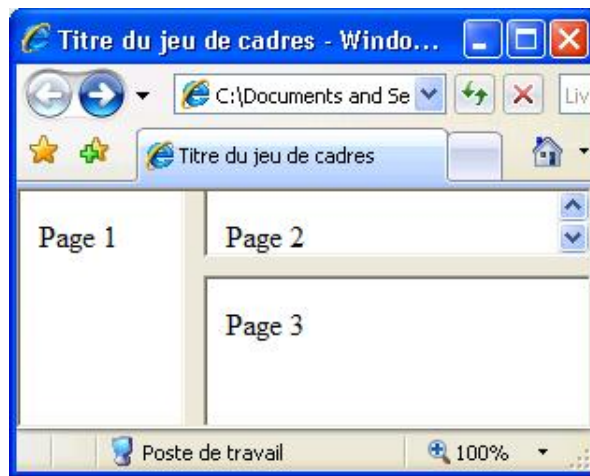
Avec l'attribut `border`, il est possible d'augmenter ou de diminuer la largeur des bordures.

La syntaxe est `border="x"` où `x` est l'épaisseur exprimée en pixels.

Exemple

```
<frameset cols="30%,70%" border="10">
<frame name="gauche" src="page1.htm">
  <frameset rows="30%,70%" border="10">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



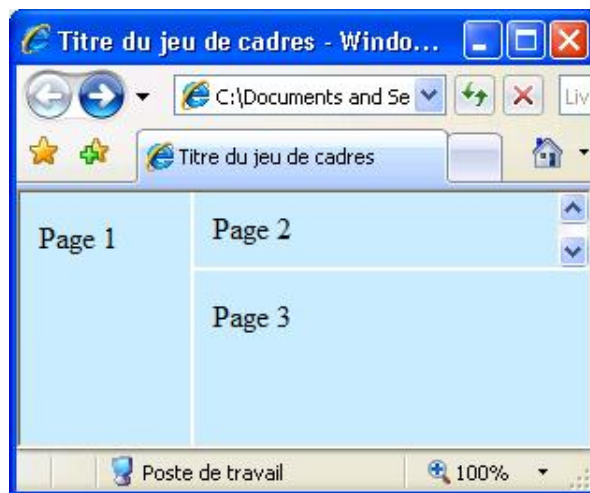
Commentaires

- L'attribut `border` peut également être affecté à la balise `<frame>`.
- L'attribut `border`, bien que parfaitement reconnu par Internet Explorer et Firefox, ne fait pas partie de la norme HTML 4.0.

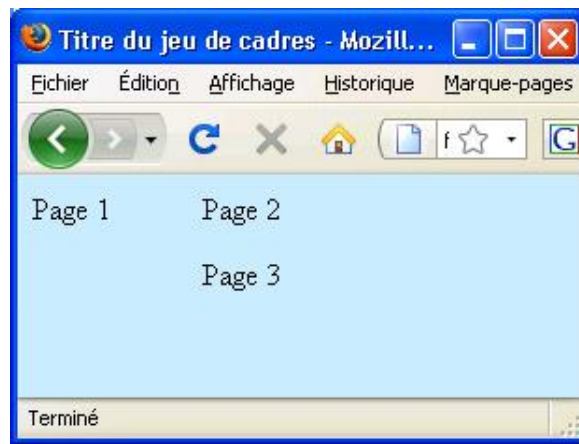
4. L'espacement entre les cadres

Tout comme pour les tableaux où il y a un espace entre les lignes et les colonnes, Internet Explorer fixe par défaut, un espace entre les cadres. Pour s'en convaincre, il suffit d'ajouter une couleur d'arrière-plan aux différents fichiers qui composent le jeu de cadres.

Résultat



Firefox n'affiche, lui, aucun espace entre les cadres.

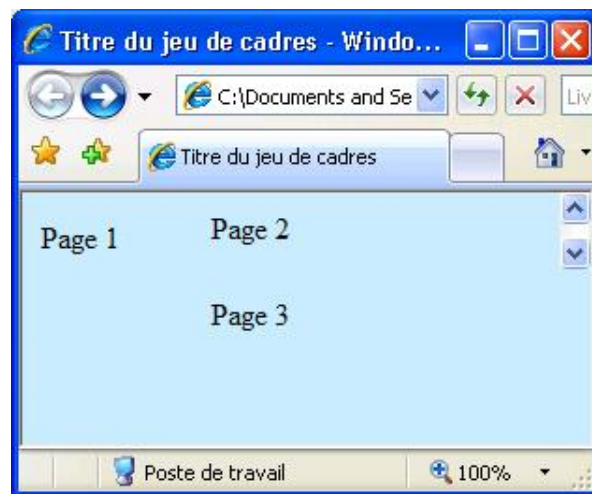


Pour modifier l'espace entre les cadres, on utilise l'attribut `framespacing="x"` où `x` est la largeur de l'espace souhaité en pixels. Ainsi pour avoir une page de cadres sans espace entre les cadres et sans bordures :

Exemple

```
<frameset cols="30%,70%" framespacing="0" frameborder="0">
<frame name="gauche" src="page1.htm">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat

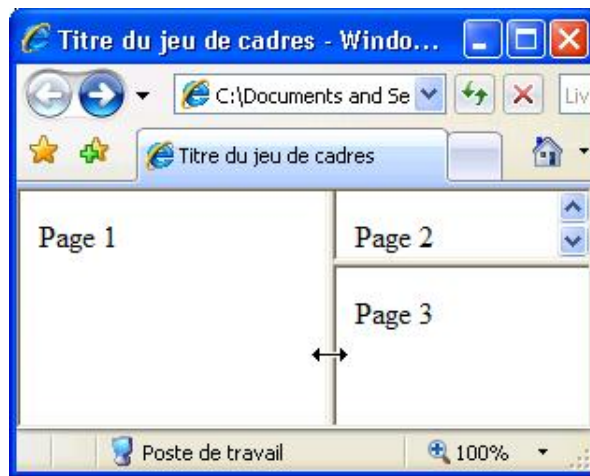


Commentaires

- Pour des raisons de compatibilité, on indique parfois les trois attributs `framespacing=0` `frameborder="0"` `border="0"`.
- L'attribut `framespacing` peut également être affecté à la balise `<frame>`.

5. Le redimensionnement des cadres

Le navigateur permet à l'internaute de modifier la grandeur des cadres ; ce que le concepteur ne souhaite généralement pas !



Pour empêcher le visiteur de redimensionner les cadres, il suffit d'ajouter l'attribut `noresize` dans la ou les balise(s) `<frame>`.

Exemple

```
<frameset cols="30%,70%">
<frame name="gauche" src="page1.htm" noresize>
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm" noresize>
    <frame name="droite_inf" src="page3.htm" noresize>
  </frameset>
</frameset>
```

Commentaire

- Il est évident que si l'attribut `frameborder="0"` et/ou `border="0"` est déjà présent, l'attribut `noresize` est sans objet.

6. La barre de défilement

Lorsque le contenu de la page affichée dans un cadre est plus grand que celui-ci, le navigateur ajoute par défaut une barre de défilement. On peut décider d'afficher cette barre de défilement en permanence, de ne jamais l'afficher ou de ne l'afficher qu'en cas de nécessité.

L'attribut à ajouter dans la balise `<frame>` est `scrolling`.

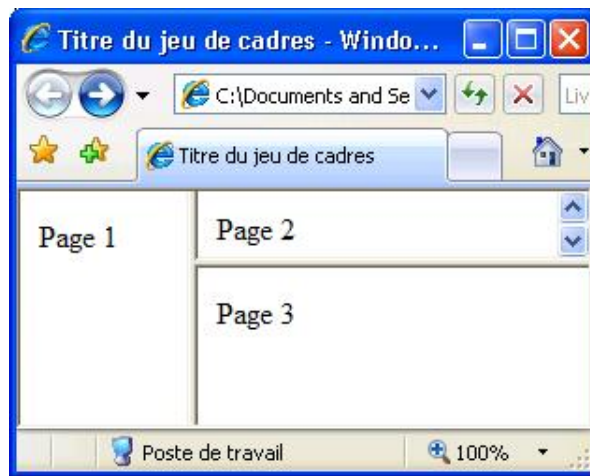
La syntaxe en est :

`scrolling="yes"` Pour afficher les barres de défilement en permanence.

`scrolling="no"` Pour ne jamais faire apparaître les barres de défilement.

`scrolling="auto"` Pour que les barres de défilement n'apparaissent qu'en cas de nécessité (par défaut).

Pour éviter la barre de défilement dans le cas présent :

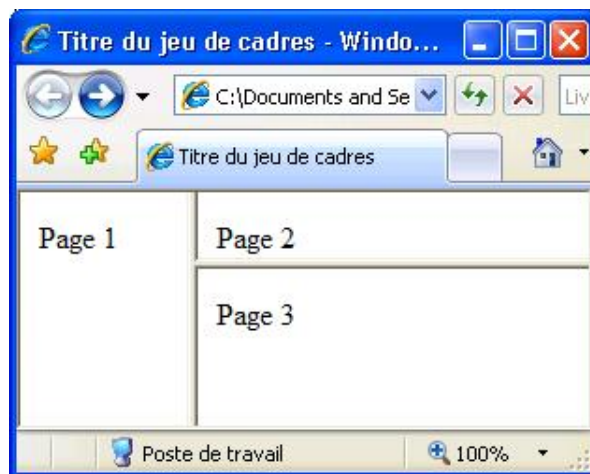


Il faut prévoir le code suivant dans la balise de frame du cadre de la page 2 pour faire disparaître cette barre de défilement :

Exemple

```
<frameset cols="30%,70%">
<frame name="gauche" src="page1.htm">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm" scrolling="no">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



Commentaire

- Un piège à éviter... Dans le cas où on applique un `scrolling="no"`, il faudra bien s'assurer que toutes les informations du cadre restent visibles, quelle que soit la résolution d'écran de votre visiteur !

7. La marge intérieure

Par défaut, le navigateur ajoute une marge de chaque côté du contenu d'un cadre. Il est possible d'augmenter mais plus fréquemment de diminuer la taille de cette marge.

On utilise l'attribut `marginwidth` et `marginheight` dans la balise `<frame>` du cadre concerné. La syntaxe de ces attributs est `marginwidth=x` pour la marge gauche/droite et `marginheight=x` pour la marge haut/bas, où `x` est un nombre exprimé en pixels.

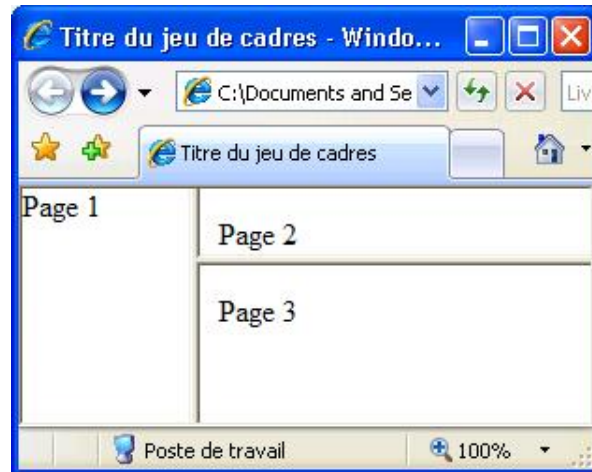
Exemple

```
<frameset cols="30%,70%">
```



```
<frame name="gauche" src="page1.htm" marginwidth="0" marginheight="0">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm" scrolling="no">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



Créer des liens entre les cadres

La construction des liens dans un jeu de cadres est nettement plus compliquée que les liens généraux abordés précédemment. En effet, en plus du lien vers le fichier souhaité, il faut déterminer la cible vers la fenêtre ou la sous-fenêtre dans laquelle devra s'afficher la page.

Cette cible peut être :

- Un cadre nommé.
- Le même cadre que celui où se situe la balise de liens.
- Une fenêtre entière (sans cadres) dans le site courant.
- Une nouvelle fenêtre qui viendra s'ouvrir parallèlement au site courant.

1. Un lien vers un cadre nommé

Lors de l'encodage de la balise `<frame>`, nous avons insisté sur l'attribut `name`, soit `name="cible"`. C'est ce nom qui sera utilisé pour déterminer la cible nommée.

Effectuez un lien dans le cadre de la page 1 (nommé "gauche") qui affichera le fichier `page4.htm` en lieu et place de `page3.htm` affiché dans le cadre nommé "droite_inf".

La syntaxe de l'attribut `cible` est ici `target="cible"`.

Dans le fichier `page1.htm`, ajoutez la balise de liens. On l'enregistre comme `page 1_2.htm`.

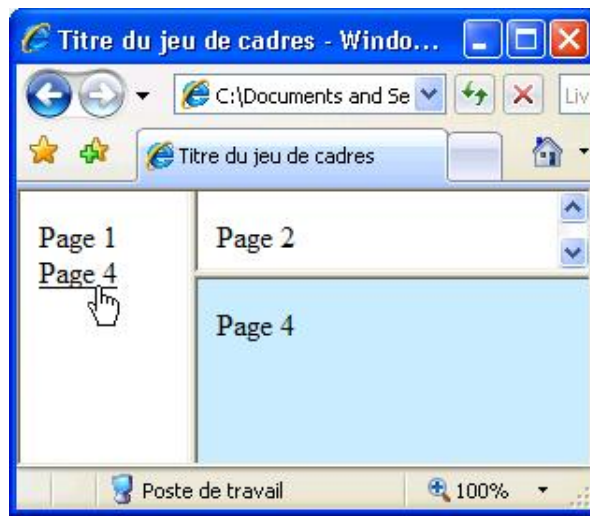
Exemple

```
<html>
<head>
<title>Page 1</title>
</head>
<body>
Page 1<br>
<a href="page4.htm" target="droite_inf">Page 4</a>
</body>
</html>
```

Le fichier de cadres devient :

```
<frameset cols="30%,70%">
<frame name="gauche" src="page1_2.htm">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



2. Un lien vers le même cadre

Toujours à partir du cadre contenant Page 1, réalisez un lien vers la page4.htm qui s'affichera dans le même cadre que celui de page1.htm (page de départ du lien).

La syntaxe de l'attribut cible est ici `target="_self"` (self pour même).

Dans le fichier page1.htm, ajoutez la balise de liens. On l'enregistre comme page1_3.htm.

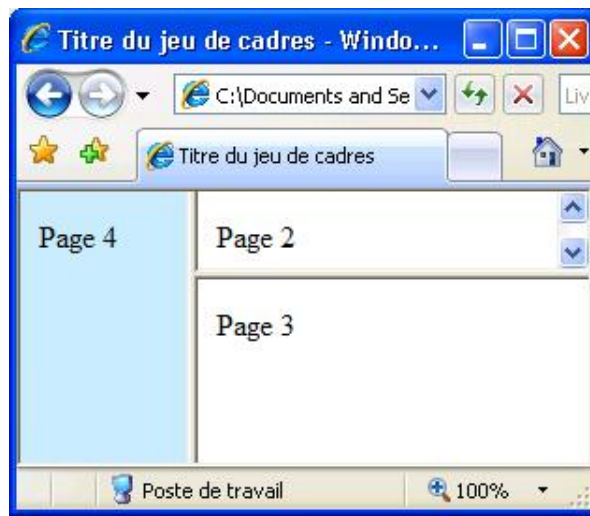
Exemple

```
<html>
<head>
<title>Page 1</title>
</head>
<body>
Page 1<br>
<a href="page4.htm" target="_self">Page 4</a>
</body>
</html>
```

Le fichier de cadres devient :

```
<frameset cols="30%,70%">
<frame name="gauche" src="page1_3.htm">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



Commentaire

- La cible nommée "gauche" aurait pu être utilisée, soit :

```
<a href="page4.htm" target="gauche">Page 4</a>
```

3. Un lien vers une fenêtre entière du site courant

Si, pour une raison ou une autre, on souhaite sortir du jeu de cadres pour afficher une page dans une fenêtre entière sans sortir du site courant, on utilisera l'attribut `target="_top"` (top pour au-dessus).

À partir du cadre contenant Page 1, réalisez un lien vers la page4.htm qui s'affichera dans une fenêtre entière.

Dans le fichier page1.htm, ajoutez la balise de liens. Le fichier devient page1_4.htm.

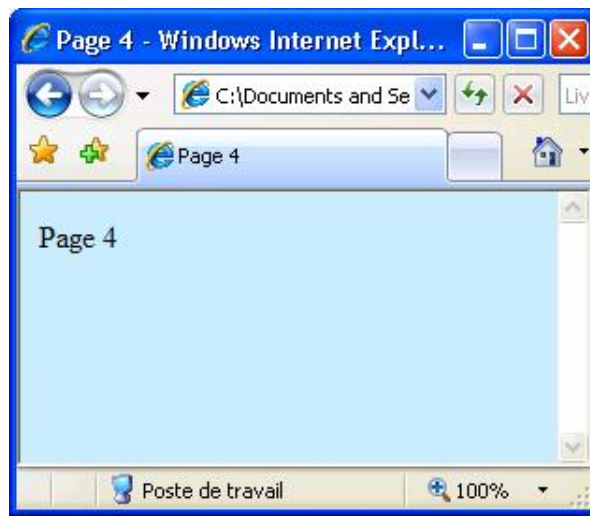
Exemple

```
<html>
<head>
<title>Page 1</title>
</head>
<body>
Page 1<br>
<a href="page4.htm" target="_top">Page 4</a>
</body>
</html>
```

Le fichier de cadres devient :

```
<frameset cols="30%,70%">
<frame name="gauche" src="page1_4.htm">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



4. Un lien vers une nouvelle fenêtre entière

Dans le cas présent, ouvrez une nouvelle fenêtre dans le navigateur. Il y a donc 2 fenêtres (ou 2 instances du navigateur) ; la fenêtre du site courant et la nouvelle fenêtre.

Un lien vers une nouvelle fenêtre du navigateur est effectué par l'attribut `target="_blank"` (blank pour page blanche ou page vierge).

Dans le fichier `page1.htm`, ajoutez la balise de liens. On l'enregistre sous `page1_5.htm`.

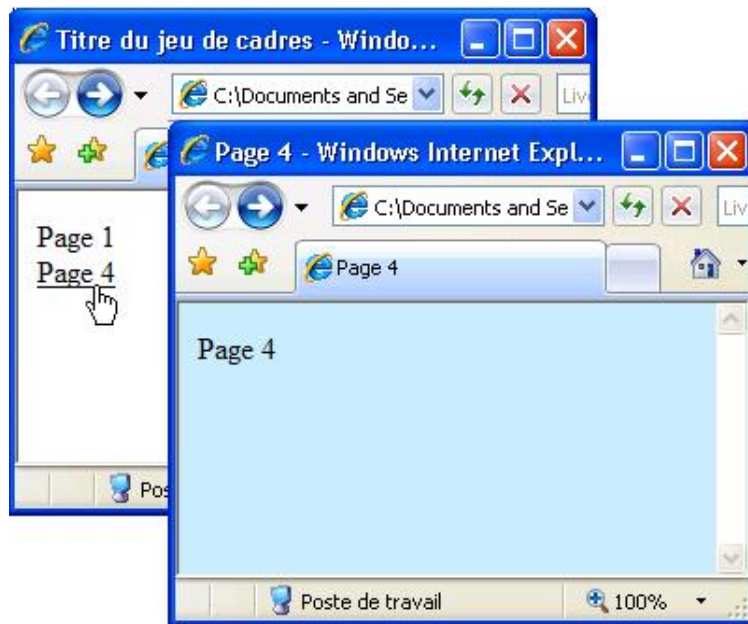
Exemple

```
<html>
<head>
<title>Page 1</title>
</head>
<body>
Page 1<br>
<a href="page4.htm" target="_blank">Page 4</a>
</body>
</html>
```

Le fichier de cadres devient :

```
<frameset cols="30%,70%">
<frame name="gauche" src="page1_5.htm">
  <frameset rows="30%,70%">
    <frame name="droite_sup" src="page2.htm">
    <frame name="droite_inf" src="page3.htm">
  </frameset>
</frameset>
```

Résultat



Résumé des targets

`target="cible"` La page désignée par le lien s'affichera dans le cadre nommé par le nom "cible" dans la balise `<frame>`.

`target="_self"` La page désignée par le lien s'affichera dans le même cadre que celui de la balise de liens de départ.

`target="_top"` La page désignée par le lien s'affichera dans une nouvelle fenêtre du site courant (sortant ainsi de la structure de cadres).

`target="_blank"` La page désignée par le lien s'affichera dans une nouvelle fenêtre ouverte par le navigateur, parallèlement au site en cours.

`target="_parent"` La page désignée par le lien s'affichera dans le cadre qui contient le frameset courant. L'effet produit diffère de celui obtenu par `_top` uniquement dans le cas de multiples framesets imbriqués.

Les cadres en ligne ou la balise iframe

La balise `<iframe>` permet d'ajouter un cadre en ligne ou un cadre flottant. Elle a toute une histoire...

À l'origine, la balise `<iframe>` était un pur produit Microsoft (Internet Explorer 3), Elle n'était donc pas affichée dans les navigateurs de la marque concurrente de l'époque, Netscape. Quelques années plus tard, cette balise propriétaire `<iframe>` a été retenue par le W3C comme standard dans ses spécifications du Html 4.0. Mais pour une raison inconnue de l'auteur, Netscape a continué à la bouder dans l'interminable série de ses versions 4. Pour ces problèmes de compatibilité entre Microsoft Explorer et Netscape, cette balise, au demeurant fort pratique, a été un peu délaissée par les créateurs de sites.

Il faut attendre la version 6 de Netscape pour qu'elle soit enfin reconnue. N'ayant plus de problèmes de compatibilité, cette balise `<iframe>` fut enfin adoptée par tous.

Firefox reconnaît bien entendu cette balise.

La balise `<iframe>` se met dans une page normale.

La syntaxe est :

```
<iframe src="page.htm" width="x" height="x"></iframe>
```

où

page.htm

est la page à afficher dans le cadre flottant.

width="x"

est la largeur du cadre exprimée en % ou en pixels.

height="x"

est la hauteur du cadre exprimée en % ou en pixels.

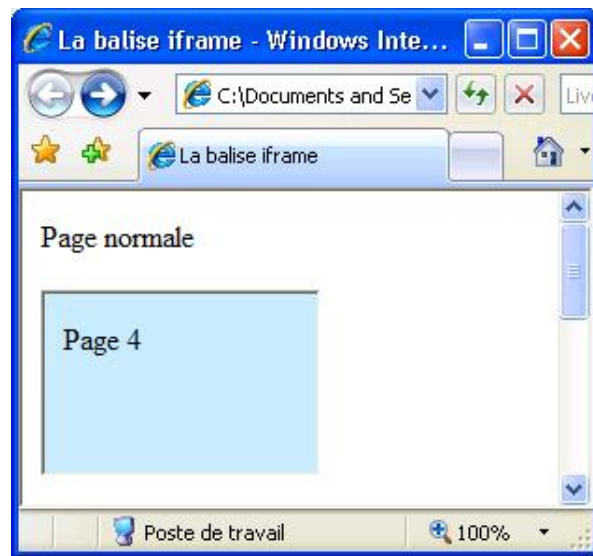
Avec la balise `<iframe>`, les attributs `frameborder`, `scrolling`, `marginwidth`, `marginheight` peuvent également être employés ; ce qui, avec toutes les possibilités de présentation de la page que contiendra le cadre, en fait une balise assez plaisante sur le plan visuel.

Insérez le fichier page4.htm dans un fichier Html conventionnel.

Exemple

```
<html>
<head>
<title>La balise iframe</title>
</head>
<body>
<p>Page normale</p>
<iframe src="page4.htm" width=150 height=100></iframe>
</body>
</html>
```

Résultat



Les cadres et le référencement par Google

Les cadres ont eu leur heure de gloire à la fin des années 90. Il faut admettre que le fait de pouvoir disposer simultanément dans la fenêtre du navigateur, les outils de navigation et le contenu, était appréciable.

Et puis, cet engouement s'est tempéré par les inconvénients inhérents à la division de la fenêtre en plusieurs cadres. Citons :

- L'imprécision de l'impression pour l'internaute débutant.
- L'impossibilité de mettre une page du jeu de cadres en favoris.
- La difficulté de retrouver l'adresse d'une page du jeu de cadres.
- Le non-respect de la séparation de la structure et du contenu si chère au W3C.

Le coup de grâce a été donné fin 2002 par une annonce de Google aux administrateurs de sites :

Nous prenons en charge les cadres "dans la mesure du possible". Les cadres peuvent en effet entraîner des problèmes avec les moteurs de recherche, les favoris (marque-pages) et l'envoi de liens par courrier électronique, car ils ne sont pas conformes au modèle de conception du Web, à savoir une URL par page. Si la requête d'un utilisateur correspond à une page dans son intégralité, Google renvoie l'ensemble des cadres (frameset) ; si elle correspond à un cadre donné de la page, Google renvoie l'URL de ce cadre. La page n'est pas présentée dans un cadre, car il n'existe peut-être pas d'ensemble de cadres correspondant à cette URL."

Il est aisément compréhensible que, pour éviter tout risque de référencement incorrect auprès du N° 1 des moteurs de recherche, les cadres sont pratiquement bannis par tous les concepteurs professionnels.

Faut-il pour autant abandonner à jamais les cadres ?...

Pourtant, des sites élaborés à partir de cadres existent encore sur le Web et profitent d'un bon référencement. La solution consiste en une utilisation judicieuse de la balise `<noframes> ... </noframes>`.

En effet, Google, un peu comme les anciens navigateurs qui ne reconnaissaient pas les cadres, lit le contenu de la balise `<noframes>`.

Soit le contenu de la balise `<noframes>` suivant :

```
<noframes>
<body>
Désolé, vous avez besoin d'un navigateur qui reconnaît les cadres pour voir ce
site.
</body>
</noframes>
```

Google retiendra pour la description du site, la phrase : "Désolé, vous avez besoin d'un navigateur qui reconnaît les cadres pour voir ce site".

Il importe, pour alimenter les outils de référencement des moteurs de recherche, de mettre entre les balises `<noframes>`, une page Web complète, avec un titre, des balises meta (voir Chapitre L'en-tête d'un document HTML), un texte d'informations et la liste des liens.

À titre d'exemple, un contenu pertinent de la balise `<noframes>` pourrait être :

```
<noframes>
<html>
<head>
<title>Html - Maîtriser le code source</title>
<meta name="description" content="Le langage Html expliqué aux débutants">
<meta name="keywords" lang="fr" content="HTML,balises,tutorial">
</head>
<body>
<h1>Maîtriser le code source</h1>
<p>
Un apprentissage progressif aux balises Html.
<p>
<a href="texte.htm">Le texte et sa présentation</a>
<a href="tableaux.htm">Les tableaux</a>
<a href="liens.htm">Les liens</a>
<a href="images.htm">Les images</a>
```

```
Etc...  
</body>  
</html>  
</noframes>
```

➤ Il faut souligner, qu'ici aussi, les feuilles de style proposent des alternatives intéressantes aux concepteurs de pages Web pour remplacer les cadres.

L'utilité des formulaires

Avec les formulaires, le Html vous ouvre les portes de l'interactivité et vous permet de recevoir des informations provenant directement de l'utilisateur et structurées selon vos souhaits.

Toute la question réside dans l'utilisation ultérieure des données !

1. Le protocole mailto

Simpliste mais pas dénué d'intérêt, l'envoi des données à une adresse électronique permet au webmestre débutant de recevoir les données d'un formulaire. Tout traitement ultérieur devra être réalisé manuellement. Ce moyen d'exploiter les formulaires présente pourtant l'énorme avantage de pouvoir être utilisé par tout un chacun sans devoir faire appel à des ressources externes.

Le protocole mailto fonctionne avec tous les navigateurs récents, depuis Explorer 4 et suivants et Firefox.

2. Le JavaScript

Vous pouvez utiliser et traiter, en interne, des données de formulaires par du JavaScript comme par exemple, passer des données d'une page à l'autre, réaliser un quiz ou vérifier les données encodées par l'utilisateur. Mais les possibilités vous sembleront bien vite limitées. Une brève découverte du JavaScript est prévue au chapitre Les autres langages du web du présent ouvrage. Vous trouverez une approche plus détaillée au chapitre 14 et suivants de l'ouvrage "Des CSS ou DHTML" dans la collection Ressources Informatiques.

3. Des applications spécifiques

Le but ultime des formulaires est bien entendu le traitement automatique des données ainsi transmises par des applications spécifiques ou des bases de données. Ces applications nécessitent d'abord de votre part des ressources de programmation que vous ne possédez peut-être pas encore. Un apprentissage de scripts CGI, Perl, PHP, C, ASP... se révélera indispensable. En outre, ces applications ne fonctionneront pas sur votre ordinateur ou celui de votre visiteur (côté client) mais sur l'ordinateur qui héberge votre site (côté serveur). Bien que les mentalités changent, pour des raisons évidentes de sécurité, les hébergeurs ne sont pas toujours désireux de faire fonctionner des exécutable qui risquent de faire "planter" leur système. Ce domaine est donc réservé aux amateurs très avertis ainsi qu'aux professionnels.

La déclaration de formulaire

Démarche habituelle en Html, il faut d'abord déclarer au navigateur que l'on va utiliser un formulaire avant d'encoder les éléments de celui-ci comme une ligne de texte et/ou des boutons à cocher. C'est la balise `<form>` qui est utilisée.

La syntaxe est :

```
<form>
... les éléments du formulaire ...
</form>
```

Cette balise comporte des attributs importants, énumérés ci-après.

1. L'attribut name

Il est parfois utile, pour un traitement ultérieur de ces formulaires, de leur prévoir un nom surtout si la page en comporte plusieurs. Utilisez l'attribut `name="nom"`.

2. L'attribut method

Les données d'un formulaire seront souvent transmises à un serveur. Cette transmission peut s'effectuer selon la méthode GET ou la méthode POST. Sans entrer dans des développements qui dépassent le cadre de cette formation, la différence entre ces deux méthodes repose sur la façon dont les données seront transmises au serveur et exploitées par celui-ci. Avec le temps, la méthode POST s'est imposée car elle apparaît plus efficace et permet le traitement d'une quantité plus importante de données. Donc si nécessaire, utilisez l'attribut `method="post"`.

3. L'attribut action

Lorsque vous donnez l'ordre au navigateur de transmettre vos données, il a besoin de savoir ce qu'il doit faire (action). En fait, il faut qu'il connaisse l'adresse de l'agent ou du "programme" à qui il doit transmettre les données du formulaire.

L'attribut de action sera :

- soit une adresse d'un script de traitement de ces données en CGI, Perl, PHP, ASP... situé sur un serveur, par exemple, `action="http:// www.serveur/cgi-bin/ma_cgi.pl"`.
- soit une adresse électronique pour récupérer simplement les données. On utilise alors le protocole mailto: suivi de l'adresse électronique de destinataire (généralement votre adresse e-mail, par exemple, `action="mailto:mon_email@serveur"`).

L'attribut action est obligatoire en Html 4.0. Afin de reproduire un code correct, nous utiliserons le subterfuge `action=""`.

4. L'attribut enctype

L'attribut enctype va spécifier sous quel format (informatique) seront transmises les données du formulaire. Par défaut, le format utilisé est le format `application/x-www-form-urlencoded`, dont le nom n'a aucune importance pour notre propos mais dont la forme est difficilement lisible. Pour se faciliter la lecture, surtout pour un envoi vers une adresse électronique avec mailto, on utilise alors `enctype="text/plain"`. Cet attribut ne peut être utilisé qu'accompagné par la méthode POST.



En cas d'oubli de la balise de fermeture `</form>`, aucun élément de votre formulaire ne sera affiché dans le navigateur.



Lorsque les données d'un formulaire sont utilisées en interne (côté client) par du JavaScript (cf. Chapitre Les autres langages du web), les attributs method, action et enctype sont inutiles car, dans ce cas, on ne fait pas appel au serveur.

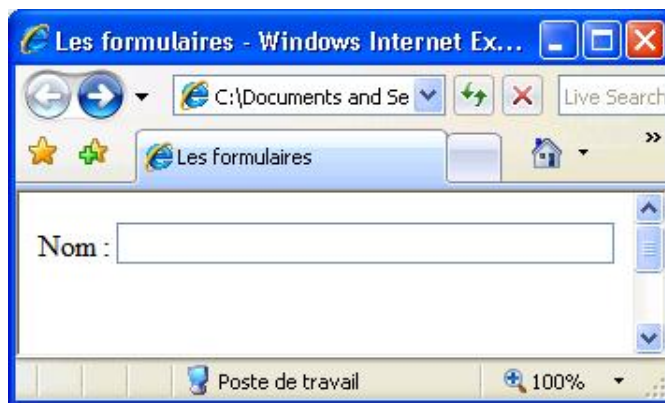
La zone de texte à une seule ligne

Simple mais fort utile, la zone de texte à une seule ligne permet de recevoir l'encodage de toutes sortes de données (lettres ou chiffres) comme le nom, l'adresse ou le code postal.

Exemple

```
<body>
<form action="">
<input type="text" size="40">
</form>
</body>
```

Résultat



Commentaires

- La balise `input` n'a pas de balise de fermeture.
- Les attributs de cette balise sont énumérés ci-après.

1. L'attribut name

Un nom d'identification peut se révéler nécessaire pour le traitement de votre formulaire par un script ou une application. L'attribut prend la forme de `name="nom"`.

2. L'attribut size

L'attribut `size="x"` où `x` est la largeur de la zone de texte, définit la taille visible de la ligne de texte.

L'utilisateur peut cependant encoder autant de caractères qu'il souhaite même s'ils débordent du champ de la zone visible à l'écran. La valeur par défaut de `size` est de 20.

3. L'attribut maxlength

L'attribut `maxlength="x"` définit le nombre maximal de caractères que l'utilisateur peut encoder dans la zone de texte.

Cet attribut est particulièrement utile pour les données de longueur définies comme par exemple 5 chiffres pour un code postal français.

4. L'attribut value

Il est possible de donner une valeur par défaut à la zone de texte par l'attribut `value="texte"` par exemple `<input type="text" value="Votre nom ici !">`.

Il est également possible de rendre ce texte en lecture seule en y ajoutant `readonly`. `<input type="text" value="Votre nom ici !" readonly>`.

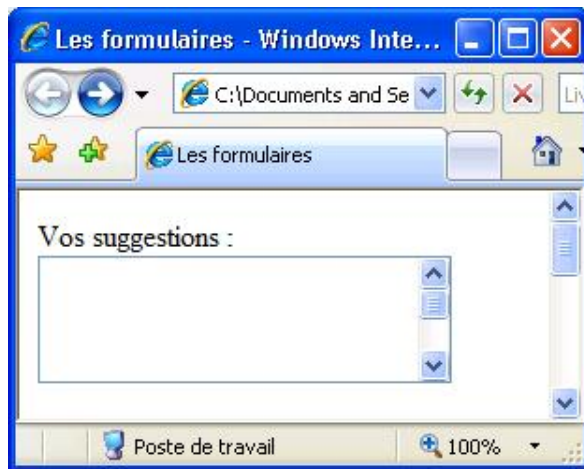
La zone de texte à plusieurs lignes

Dans certains cas, il faut prévoir plus d'espace à l'utilisateur pour encoder par exemple ses remarques et suggestions. On utilise dans ce cas la balise `<textarea>...</textarea>`.

Exemple

```
<body>
<form action="">
Vos suggestions :<br>
<textarea rows="4" cols="25"></textarea>
</form>
</body>
```

Résultat



Commentaire

- La balise `<textarea>` comporte une balise de fermeture `</textarea>`.

1. L'attribut name

L'attribut `name="nom"` permet de récupérer les données de cet élément ainsi nommé.

2. Les attributs rows et cols

L'attribut `rows="x"` permet de déterminer la hauteur visible de la zone de texte.

L'attribut `cols="x"` permet de déterminer la largeur réservée aux caractères dans la zone de texte. Ici aussi le texte peut être plus long que la zone affichée.

3. L'attribut wrap

L'attribut `wrap` permet le retour à la ligne automatique du texte lorsque le bord droit de la zone est atteint. Le retour à la ligne automatique est effectué par défaut par Explorer et Firefox.

Avec `wrap="virtual"` le passage à la ligne se fait automatiquement lors de la saisie du texte mais lors de l'envoi du formulaire, aucun caractère de changement de ligne n'est cependant transmis.

Avec `wrap="physical"` le passage à la ligne se fait aussi automatiquement mais avec un caractère de passage à la ligne.



Cet attribut `wrap` ne fait pas partie des attributs du HTML 4.0. Il peut dès lors présenter quelques soucis de compatibilité.

4. La valeur par défaut

Une valeur par défaut peut être prévue dans la zone de texte multiligne, non pas par l'attribut `value` mais de la façon suivante : `<textarea rows="4" cols="25">Vos suggestions ici! </textarea>`. L'attribut `read only` peut aussi être appliqué.

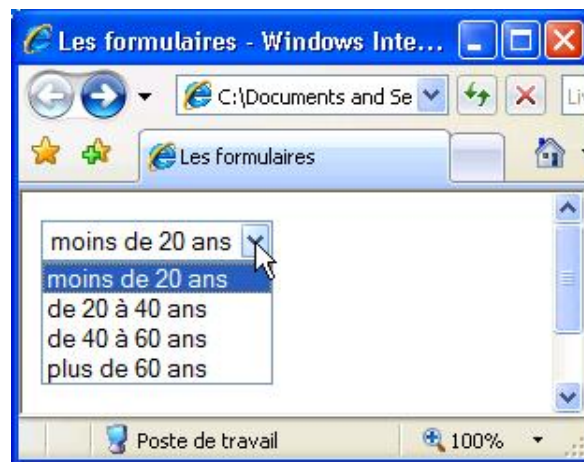
Le menu déroulant

Le menu déroulant ou la liste de choix est un autre élément de formulaire.

Exemple

```
<body>
<form action="">
<select>
<option>moins de 20 ans</option>
<option>de 20 à 40 ans</option>
<option>de 40 à 60 ans</option>
<option>plus de 60 ans</option>
</select>
</form>
</body>
```

Résultat



Commentaire

- La balise `<select>... </select>` indique au navigateur l'usage d'une liste déroulante. Les éléments de la liste sont introduits par la balise `<option> ... </option>`.

1. L'attribut name

Cet attribut `name="nom"`, appliqué à la balise `<select>`, permet de donner un nom à l'élément en vue d'un traitement ultérieur.

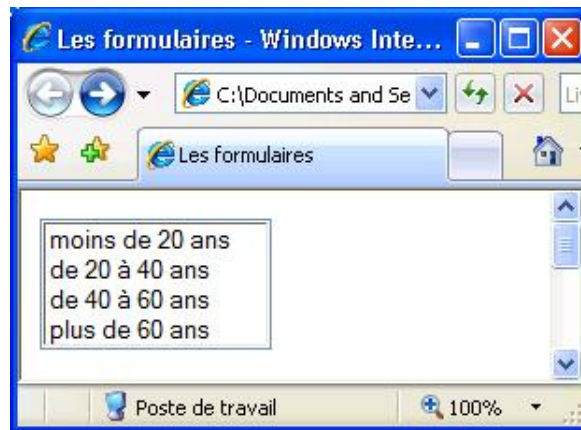
2. L'attribut size

Par défaut, l'attribut `size` de la balise `<select>` est égal à 1, ce qui est assez pratique car cela permet un gain de place appréciable dans la disposition de la page. Cependant, par l'attribut `size="x"`, vous pouvez définir le nombre d'éléments du menu qui sera visible. Votre menu n'aura alors plus rien de déroulant !

Exemple

```
<body>
<form action="">
<select size="4">
<option>moins de 20 ans</option>
<option>de 20 à 40 ans</option>
<option>de 40 à 60 ans</option>
<option>plus de 60 ans</option>
</select>
</form>
```

Résultat



3. L'attribut multiple

Par défaut, l'utilisateur ne peut utiliser qu'un choix du menu déroulant. Avec l'attribut multiple de la balise <select>, plusieurs choix peuvent être effectués. Pour ce faire, l'utilisateur doit maintenir la touche [Ctrl] du clavier et cliquer sur les éléments avec la souris. Il est alors préférable de rappeler dans la page cette façon de procéder, peu commune pour l'internaute moyen.

L'attribut `size` doit être spécifié et égal au nombre des balises <option>.

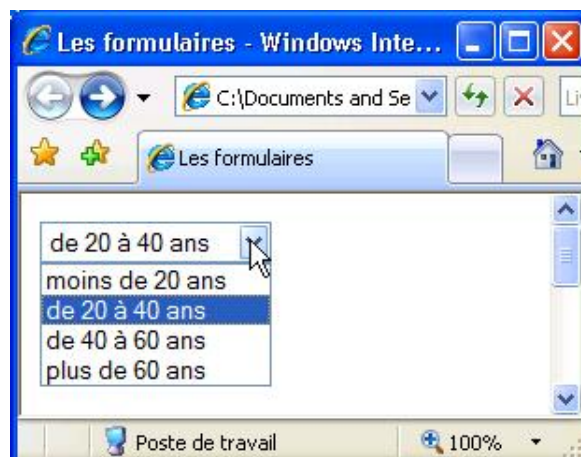
4. L'attribut selected

Par défaut, le premier élément de la liste est retenu. Mais il est possible de présélectionner un autre élément par l'attribut `selected` de la balise <option>.

Exemple

```
<body>
<form action="">
<select>
<option>moins de 20 ans</option>
<option selected> de 20 à 40 ans</option>
<option> de 40 à 60 ans</option>
<option>plus de 60 ans</option>
</select>
</form>
</body>
```

Résultat



5. L'attribut value

En principe, c'est le texte de l'élément choisi, placé derrière `<option>` qui est transmis lors de l'envoi du formulaire. Vous pouvez toutefois spécifier qu'une autre valeur, généralement chiffrée, soit transmise avec l'attribut `value="valeur"`.

Exemple

```
<body>
<form action="">
<select>
<option value="1">moins de 20 ans</option>
<option value="2">de 20 à 40 ans</option>
<option value="3">de 40 à 60 ans</option>
<option value="4">plus de 60 ans</option>
</select>
</form>
</body>
```

Commentaire

- Si la première option était retenue, ce n'est plus "moins de 20 ans" qui serait transmis mais simplement la valeur 1.

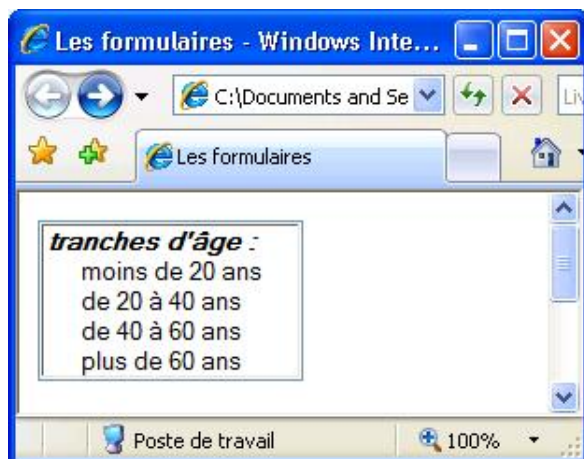
6. La balise `<optgroup>`

La balise `<optgroup label="titre...du_menu"> ... </optgroup>` permet de structurer le menu. On entoure les éléments du sous-menu par la balise `<optgroup> ... </optgroup>`. L'attribut `label` donne un titre à ce menu.

Exemple

```
<body>
<form action="">
<select size="5">
<optgroup label="tranches d'âge :">
<option>moins de 20 ans</option>
<option> de 20 à 40 ans</option>
<option> de 40 à 60 ans</option>
<option>plus de 60 ans</option>
</optgroup>
</select>
</form>
</body>
```

Résultat



Commentaire

- Pour les utilisateurs d'Internet Explorer, cette balise n'est correctement affichée qu'à partir de la version 6.

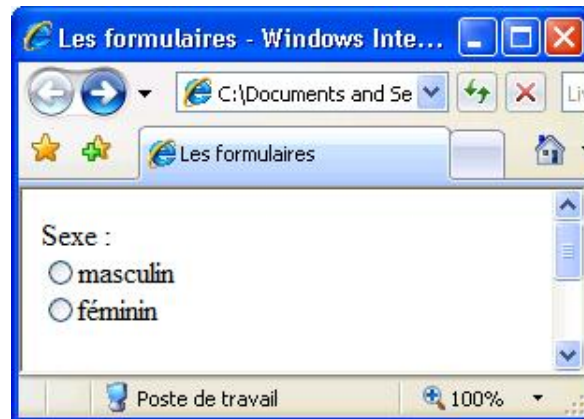
Les boutons de choix unique (radio)

Les boutons d'option, aussi appelés boutons radio, ont comme particularité qu'une seule option à la fois peut être activée (le "ou" exclusif).

Exemple

```
<body>
<form action="">
Sexe :<br>
<input type= "radio" name="sexe">masculin<br>
<input type= "radio" name="sexe">féminin
</form>
</body>
```

Résultat



Commentaire

- La balise `<input>` n'a pas de balise de fermeture.

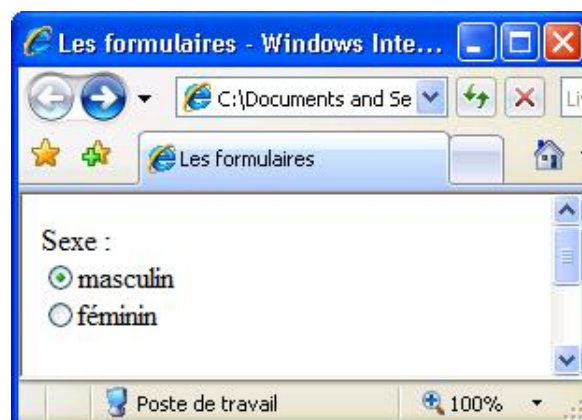
1. L'attribut name

Ici l'attribut name est **obligatoire** ! En outre, dans le cas de boutons radio, le nom doit être identique pour tous les boutons.

2. L'attribut checked

Il est possible d'activer un bouton par défaut à l'ouverture de la page par l'attribut `checked`.

Ainsi dans l'exemple précédent `<input type= "radio" name= "sexe" checked> masculin
` aurait le résultat suivant.



3. L'attribut value

En vue d'un traitement ultérieur, on peut bien entendu attribuer une valeur à chaque bouton par l'attribut `value="valeur"`.

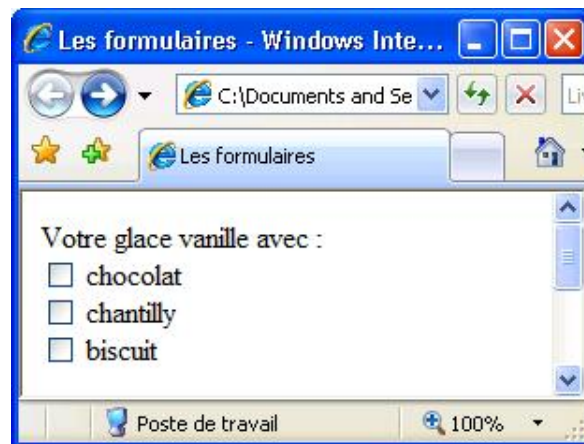
Les boutons de choix multiples (checkbox)

La mise en œuvre de ces boutons de choix multiples est proche des boutons à choix unique mais à la différence que, dans le cas présent, plusieurs choix simultanés peuvent être réalisés.

Exemple

```
<body>
<form action="">
Votre glace vanille avec :<br>
<input type="checkbox" name="nom0"> chocolat<br>
<input type="checkbox" name="nom1"> chantilly<br>
<input type="checkbox" name="nom2"> biscuit<br>
</form>
</body>
```

Résultat



Commentaire

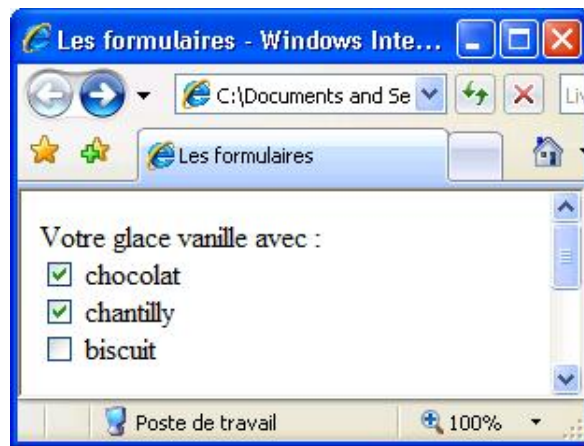
- La balise `<input>` n'a pas de balise de fermeture.

1. L'attribut name

L'attribut `name="nom"` est **obligatoire**. Les règles sont moins précises que pour les boutons d'option. Vous pouvez employer des noms identiques ou des noms différents pour chaque case à cocher. Cependant des noms différents sont indispensables pour leur utilisation dans un script.

2. L'attribut checked

Vous pouvez activer par défaut un bouton ou plusieurs boutons à l'ouverture de la page par l'attribut `checked`. Ainsi dans l'exemple précédent, `<input type="checkbox" name="nom0"> chocolat` et `<input type="checkbox" name="nom1"> chantilly` ont été présélectionnés. Le résultat est le suivant :



3. L'attribut value

En vue d'un traitement ultérieur, on peut bien entendu attribuer une valeur à chaque bouton par l'attribut `value="valeur"`.

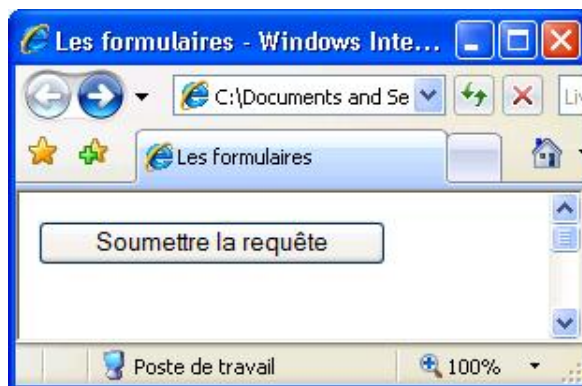
Le bouton d'envoi

Par le bouton d'envoi, appelé parfois aussi bouton de soumission, les données du formulaire seront transmises selon l'attribut `action` défini dans la balise `<form>`. Il est donc indispensable pour l'utilité de votre formulaire.

Exemple

```
<body>
<form action="">
<input type="submit">
</form>
</body>
```

Résultat



Commentaires

- La balise `<input>` n'a pas de balise de fin.
- Par défaut, le texte du bouton est celui choisi par votre navigateur (citons par exemple "Envoyer", "Soumettre" ou "Submit"). L'exemple ci-dessus est celui de Explorer 7. Il est cependant possible de modifier le texte du bouton par défaut par l'attribut `value="Votre propre texte"`. Firefox est plus conventionnel avec "Envoyer".



- Ces boutons standards du Html ne permettent pas beaucoup de fantaisie ou de créativité. La version 4.0 du Html a introduit la possibilité de créer ses propres boutons et surtout d'y adjoindre des images.
- On utilise alors une autre balise, la balise `<button type="submit"> ... </button>`, où tout ce qui se trouve entre la balise d'ouverture et de fermeture sert de bouton. Cela peut être du texte (que vous pourrez formater avec des feuilles de style) mais également une image. La balise devient alors : `<button type="submit"></button>`.

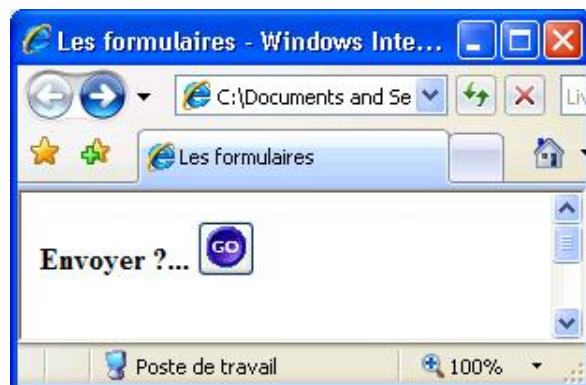
Exemple

Remplacez le bouton d'envoi traditionnel par une image (ici `go.gif` située dans le même dossier). Le code devient :

```
<body>
```

```
<form action="">
Envoyer ?...
<button type="submit"></button>
</form>
</body>
```

Résultat



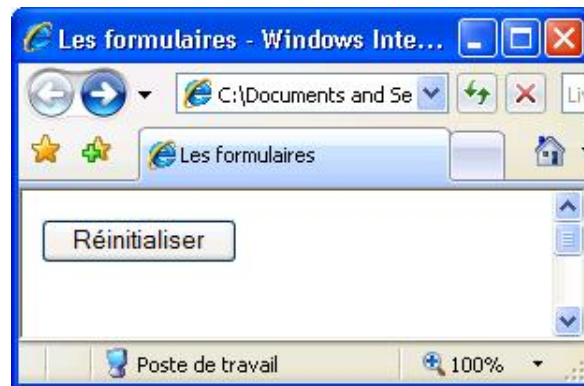
Le bouton d'annulation

Il est opportun de permettre à l'utilisateur d'annuler son encodage et de réinitialiser (reset) un formulaire vierge.

Exemple

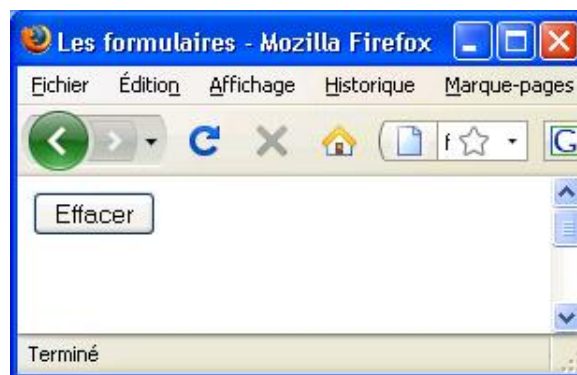
```
<body>
<form action="">
<input type="reset">
</form>
</body>
```

Résultat



Commentaires

- La balise `<input>` n'a pas de balise de fin.
- Ici aussi vous pouvez modifier le texte par défaut par l'attribut `value="Votre texte"`.
- Dans le même ordre d'idée que pour le bouton d'envoi, le Html 4.0 a introduit une balise pour réaliser cette fonction à partir d'une image. C'est la balise `<button type="reset"> ... </button>` qui avec une image devient : `<button type="reset"></button>`
- Ici aussi le texte par défaut peut varier d'un navigateur à l'autre. Si Internet Explorer 7 utilise le terme "Réinitialiser", Firefox annonce "Effacer" qui est plus parlant.



Le bouton de commande

Il est également possible de créer des boutons dont l'action spécifique sera définie par le webmestre, généralement en faisant appel à du JavaScript (cf. Chapitre Les autres langages du Web).

On définit par exemple, une balise `<button>` de type="button" (car il y a aussi le type submit et reset). L'action du bouton est déclenchée par un événement JavaScript. Cet événement est ici `onClick`, soit au clic de la souris qui entraînera l'ouverture de la page dont la localisation est le fichier "suite.htm".

Exemple

```
<body>
<form action="">
<button type="button" onClick="location.href='suite.htm'">
 Suite
</button>
</form>
</body>
```

Résultat



Commentaire

- Tout le contenu, texte et/ou image, des balises `<button>` est repris dans le bouton.

Les formulaires cachés (hidden)

Les formulaires cachés permettent de stocker des données qui ne seront pas visibles par le visiteur de la page. Ce genre de formulaire fait le régal des programmeurs confirmés (spécialement en JavaScript) car il peut contenir des données d'un formulaire précédent, des données brutes ou des données issues d'un script. Tout ce qui est caché est bien mystérieux !

Exemple

```
<body>
<form action="">
<input type="hidden" value="données_cachées">
</form>
</body>
```

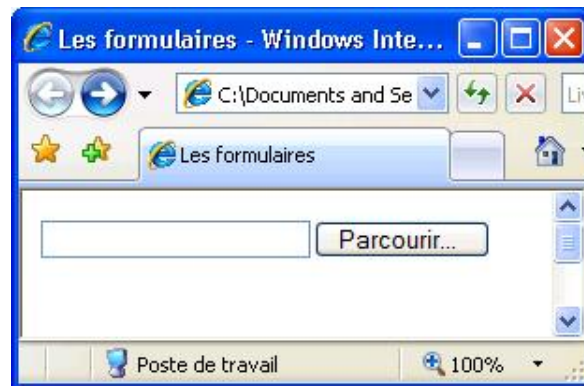
Les formulaires de transfert de fichier

Les formulaires de transfert de fichier (file) permettent à l'utilisateur de transmettre avec un formulaire, un fichier de son ordinateur local vers un ordinateur de type serveur.

Exemple

```
<body>
<form action="..." method="post" enctype="multipart/form-data">
<input type="file">
</form>
</body>
```

Résultat



Commentaires

- Dans la déclaration de la balise `<form>`, il faut impérativement utiliser la méthode `post` et `enctype="multipart/form-data"` pour le transfert du fichier au bon format.
- Les attributs `name`, `size` et `maxlength` (pour la taille maximale du fichier) peuvent être appliqués et ne devraient plus poser de problèmes à ce stade de l'étude.
- Par contre l'attribut `accept="..."` est assez intéressant car il permet de limiter le transfert à certains types de fichier. Comme type de fichiers, citons des fichiers texte (txt), word (doc), excel (xls), pdf, htm, etc. Dans la désignation des fichiers, les jokers "*" sont acceptés.
- En fait, cette balise ne sert qu'à sélectionner le fichier à transférer. Le transfert lui-même sera pris en charge par une application côté-serveur, comme le PHP.

Les formulaires de mot de passe

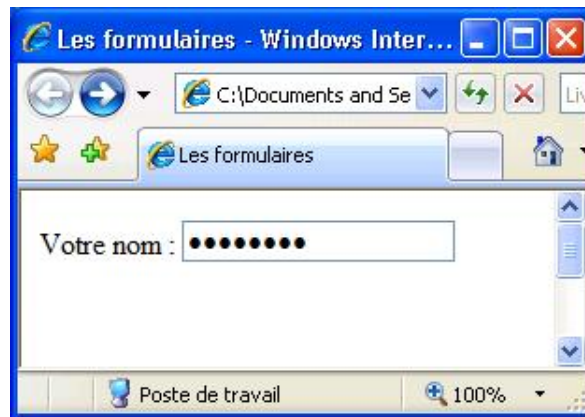
Ce type de formulaire est en fait une simple zone de texte mais dont l'encodage est remplacé, à l'affichage, par des puces ou des astérisques.

Ce formulaire de mot de passe ne protège en aucun cas les données, car elles seront transmises en clair. Elles vous protègent uniquement contre les personnes qui pourraient vous lire durant l'encodage.

Exemple

```
<body>
<form action="">
Votre nom :
<input type="password">
</form>
</body>
```

Résultat



Commentaire

- Les attributs de la zone de texte `name`, `size` et `maxlength` peuvent être utilisés.

L'organisation des éléments d'un formulaire

1. Les balises <fieldset> et <legend>

Dans le cas de formulaires longs et complexes, il est parfois utile de regrouper graphiquement certains éléments pour organiser la page et en améliorer la lisibilité.

La balise <fieldset> ... </fieldset>, qui se place après la balise <form>, regroupe les éléments du champ que vous déterminez. Le champ sera alors visualisé à l'écran par une bordure.

La balise <legend>Texte de la légende</legend> fournit la légende qui vient s'insérer dans le cadre dessiné par <fieldset>. Cette balise admet l'attribut align="left" (défaut) et align="right".

Exemple

```
<body>
<form action="">
<fieldset>
<legend>Coordonnées</legend>
Nom : <input type="text" size="25"><br>
Prénom : <input type="text" size="25"><br>
E-mail : <input type="text" size="25">
</fieldset>
<fieldset>
<legend>Suggestions</legend>
<textarea cols="25" rows="3"></textarea>
</fieldset>
</form>
</body>
```

Résultat



2. La balise <label>

La balise <label> associe explicitement l'intitulé à un contrôle de formulaire particulier. Un peu comme si on collait une étiquette (label) en face d'un élément de formulaire.

La balise <label> et son contenu n'ont aucun effet visible à l'écran mais elle est particulièrement utile dans le domaine de l'accessibilité des sites Web aux personnes non-voyantes. Cette balise <label> est prise en charge par les aides techniques et facilite grandement l'utilisation des formulaires par les personnes atteintes d'un handicap visuel.

Dans un premier temps, le texte assigné à un élément de formulaire doit être placé entre les balises `<label>` ... `</label>`.

```
<label>Nom</label> :  
<input type="text"><br>  
<label>Prénom</label> :  
<input type="text"><br>  
<label>Adresse email</label> :  
<input type="text"><br />  
<label>Bouton d'envoi</label> :  
<input type="button" value="Envoyer">
```

Il faut ensuite relier cette étiquette label au contrôle de formulaire. Pour ce faire, l'élément de formulaire sera défini par un identifiant de type *id* (voir la partie consacrée aux feuilles de style).

```
<label>Nom</label> :  
<input type="text" id="f1"><br>  
<label>Prénom</label> :  
<input type="text" id="f2"><br>  
<label>Adresse email</label> :  
<input type="text" id="f3"><br />  
<label>Bouton d'envoi</label> :  
<input type="button" value="Envoyer" id="f4">
```

L'attribut `for="..."` ajouté à la balise `<label>` permet de raccorder directement l'étiquette à l'élément de formulaire en faisant référence à cet identifiant.

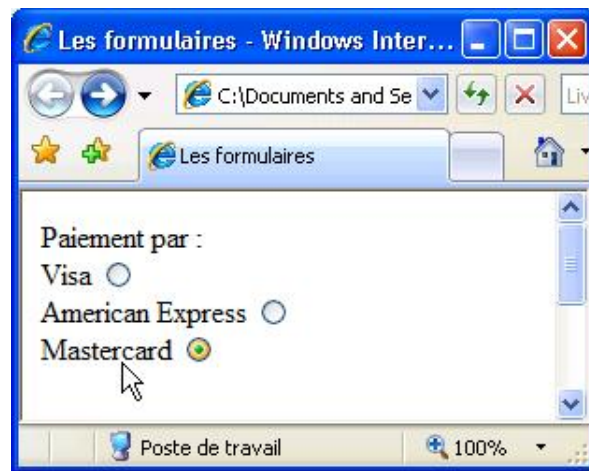
```
<label for="f1">Nom</label> :  
<input type="text" id="f1"><br>  
<label for="f2">Prénom</label> :  
<input type="text" id="f2"><br>  
<label for="f3">Adresse email</label> :  
<input type="text" id="f3"><br />  
<label for="f4">Bouton d'envoi</label> :  
<input type="button" value="Envoyer" id="f4">
```

Il sera alors possible de sélectionner un élément de formulaire par un simple clic sur le texte qui lui est joint. Ce qui est très apprécié par les non-voyants et assez ergonomique, même pour les utilisateurs voyants.

Exemple

```
<body>  
<form action="">  
<p>Paielement par :<br>  
<label for="cash1">Visa</label>  
<input type="radio" name="cash" id="cash1"><br>  
<label for="cash2">American Express</label>  
<input type="radio" name="cash" id="cash2"><br>  
<label for="cash3">Mastercard</label>  
<input type="radio" name="cash" id="cash3">  
</p>  
</form>  
</body>
```

Résultat



La balise `<label>` peut s'appliquer à tous les différents éléments de formulaire.

Les formulaires et les feuilles de style

Les feuilles de style permettent de personnaliser, voire d'égayer les formulaires. Citons :

- appliquer une couleur aux boutons,
- changer la couleur du bouton lors du survol de la souris,
- ajouter un arrière-plan de couleur à une ligne ou zone de texte, à une liste de sélection,
- etc.

Un exemple complet : un livre d'or

Il est possible d'ajouter un livre d'or (*guestbook*) à votre site.

Par expérience, on remarque cependant que les visiteurs sont assez réticents pour encoder leurs coordonnées personnelles sur le Web. N'espérez pas d'adresse postale ou de numéro de téléphone, car on reste finalement assez sensible à son anonymat sur le Web.

Voici donc, comme exemple final, un livre d'or avec le nom, le prénom, l'adresse, le code postal (maximum 5 caractères), la ville et l'adresse e-mail.

Exemple

```
<html>
<head>
<title>Mon livre d'or</title>
</head>
<body>
<form method="post" action="mailto:votre_adresse_Email"
enctype="text/plain">
Votre nom :<br>
<input type="text" name="nom"><br>
Votre prénom :<br>
<input type="text" name="prenom"><br>
Votre adresse :<br>
<textarea name="adresse" rows="2" cols="27"></textarea><br>
Code Postal :<br>
<input type="text" name="code_postal" size="5" maxlength="5"><br>
Ville :<br>
<input type="text" name="ville"><br>
Adresse e-mail :<br>
<input type="text" name="email" size="35">
<p>
<input type="submit" value="Envoyer">
<input type="reset" value="Annuler">
</p>
</form>
</body>
</html>
```

Résultat

Mon livre d'or - Windows Intern...

C:\Documents and Se

Mon livre d'or

Votre nom :

Votre prénom :

Votre adresse :

Code Postal :

Ville :

Adresse e-mail :

Envoyer Annuler

Poste de travail 100%

Commentaire

- À la réception, le courrier électronique aura la forme suivante :

nom=Ma Cousine

prenom=Bécassine

adresse=Place Goya, 52

code_postal=87600

ville=Yséploue

mail=macousinebecassine@free.fr

Soit à chaque fois le nom donné à l'élément de formulaire, le signe égal = et les données encodées par l'utilisateur.

Introduction

Un document Html est, comme évoqué au chapitre Présentation, composé de deux parties :

1. L'en-tête, compris entre les balises `<head> . . . </head>`
2. Le corps du document compris entre les balises `<body> . . . </body>`.

Généralement, c'est le corps du document qui retient l'attention des programmeurs Html débutants. Le contenu des balises `<head>` est très souvent négligé, voire ignoré.

Pourtant, les balises d'en-tête sont capitales pour faire de votre document, un document performant auprès des moteurs de recherche et donc pour générer des visites de vos pages.

Le DOCTYPE et la validation

Le DOCTYPE nous rappelle que le langage Html est dérivé du SGML où le DTD est indispensable. Nous reviendrons sur le DOCTYPE et la validation au chapitre 12 lors de l'étude du XHTML.

1. Le DOCTYPE

Sur la toute première ligne d'un fichier Html (soit avant la balise `<html>`), on retrouve l'instruction DOCTYPE qui permet de définir la version du langage Html utilisée dans la page.

Cette déclaration, avec le mode de fonctionnement des navigateurs actuels, est facultative. Cependant, elle est fortement conseillée car elle force le navigateur à respecter la norme référencée, ce qui corrige parfois certains problèmes de compatibilité. En outre, le DOCTYPE est nécessaire pour les validateurs Html [validators] afin qu'ils puissent savoir, selon quelle version du Html, ils devront valider ou "corriger" le code de la page.

Version	Déclaration
HTML 2.0	<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 2.0//EN"></code>
HTML 3.2	<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN"></code>
HTML 4.0 (transitoire)	<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" "http://www.w3.org/TR/REC-html40/loose.dtd"></code>
HTML 4.0 (strict)	<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN""http://www.w3.org/TR/REC-html40/strict.dtd"></code>
HTML 4.0 (frameset)	<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN" "http://www.w3.org/TR/REC-html40/frameset.dtd"></code>

Notons que les éléments de la déclaration DOCTYPE sont "case sensitive". Respectez donc les majuscules et minuscules !

Dans le cadre de cet ouvrage, on s'intéressera plus particulièrement aux DOCTYPE du Html 4.0.

Ceux-ci comportent 3 versions :

- Le doctype strict qui comme son nom l'indique bien est très rigoureux sur le plan du code. Il est assez difficile, surtout pour un débutant, de respecter scrupuleusement cette norme car elle ne reprend pas toute une série de balises et d'attributs.
- Le doctype Transitional qui est recommandé si l'on souhaite bénéficier de plus de libertés dans la conception du code Html. C'est celui que nous avons retenu dans le cadre de notre présente étude.
- Le doctype Frameset qui est réservé aux cadres (cf. Chapitre Les cadres).

Nous avons jusqu'à présent volontairement ignoré toute référence au DOCTYPE dans nos différents exemples pour ne pas surcharger notre approche du Html et proposer un code plus simple à appréhender. Le DOCTYPE et le respect des normes qu'il induit sont cependant des éléments essentiels de la conception des documents Html.

2. La validation

Un document Html conforme aux standards du W3C et en accord avec le DOCTYPE retenu, est l'assurance d'un code parfait qui évitera les problèmes d'affichage dans les différents navigateurs actuels.

Il est ainsi vivement conseillé de valider le code de la page.

a. Les validateurs en ligne

Le Validateur du W3C

Le W3C lui-même présente un validateur de code (validator.w3.org/). Ce qui ne peut être qu'un gage de qualité.



Notons :

- Qu'il est en anglais.
- Qu'il offre de valider un fichier par son adresse http, un fichier présent sur votre ordinateur ou du code inséré dans une zone de texte.
- Qu'il est très pointilleux.
- Que ses remarques et explications sont très techniques voire parfois sibyllines.
- Qu'il est "la" référence des professionnels.
- Qu'il est celui retenu pour valider les exemples de ce livre.

Le validateur du W3Québec

Le W3Québec est un organisme sans but lucratif visant à promouvoir l'ensemble des normes, standards ouverts et bonnes pratiques du Web et du multimédia.



Il propose un validateur (w3qc.org/valideur/) :

- En français
- Avec la possibilité de valider un fichier par son adresse http, un fichier présent sur votre ordinateur ou du code inséré dans une zone de texte.
- Les avertissements ou erreurs sont fournis en français dans un langage technique mais accessible.
- Il manque parfois des explications plus explicites pour corriger le code soumis.
- Le niveau de qualité semble être satisfaisant.

Il est peut-être intéressant d'en voir les résultats sur quelques lignes de code.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
</head>
<body>
```

```
Texte <b><i>Gras et italique </b></i>
</body>
</html>
```

Mode 3

Insérez le code source à vérifier :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
</head>
<body>
Texte<b><i>Gras et italique</b></i>
</body>
```

[\(Mode 3 avec plus d'options\)](#)

Après validation, le validateur nous apprend que le document comporte 6 erreurs et qu'il n'est donc pas valide.

```
Nombre d'erreurs : 3.
Nombre d'avertissements : 0.
Nombre d'erreurs et avertissements différents : 3.
Nombre de lignes : 10.
Nombre d'erreurs par ligne : 0.30.
Nombre de lignes erronées : 2.
Passage : Cette page est invalide selon le DOCTYPE utilisé.
```

Il détaille ensuite le code comme suit :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd"> <html>
<head>
</head>
Erreur à la colonne 6 : fin d'étiquette pour "HEAD" lequel n'est pas
complété
La balise fermante </head> n'est pas permise ici car il manque la balise
de titre <title> ... </title> obligatoire.
<body>
Texte <b><i>Gras et italique</b></i>

Erreur à la colonne 30 : fin d'étiquette pour "I" est omise mais sa
déclaration ne permet pas cela (type #68)
Erreur à la colonne 34: fin d'étiquette pour l'élément "I" lequel n'est
pas ouvert (type #79)

Les balises <b> et </i> sont mal imbriquées.

</body>
</html>
```

Le code corrigé sera jugé valide.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Titre</title>
</head>
```

```
<body>
Texte <b><i>Gras et italique</i></b>
</body>
</html>
```

Validome

Validome (www.validome.org/) est un logiciel de validation non commercial, offrant aux webmasters ambitieux et aux développeurs professionnels un outil rapide et sûr permettant de vérifier si leurs documents HTML / XHTML / XML ou leurs sites sont conformes aux standards actuels approuvés par le W3C.



Du même acabit que les précédents. Regrettons cependant que la possibilité d'insertion du code dans une zone de texte n'existe pas et que la validation ne puisse se réaliser qu'en fournissant l'adresse de la page ou en téléchargeant le fichier à valider.

b. Les validateurs en local

Html Validator pour Firefox

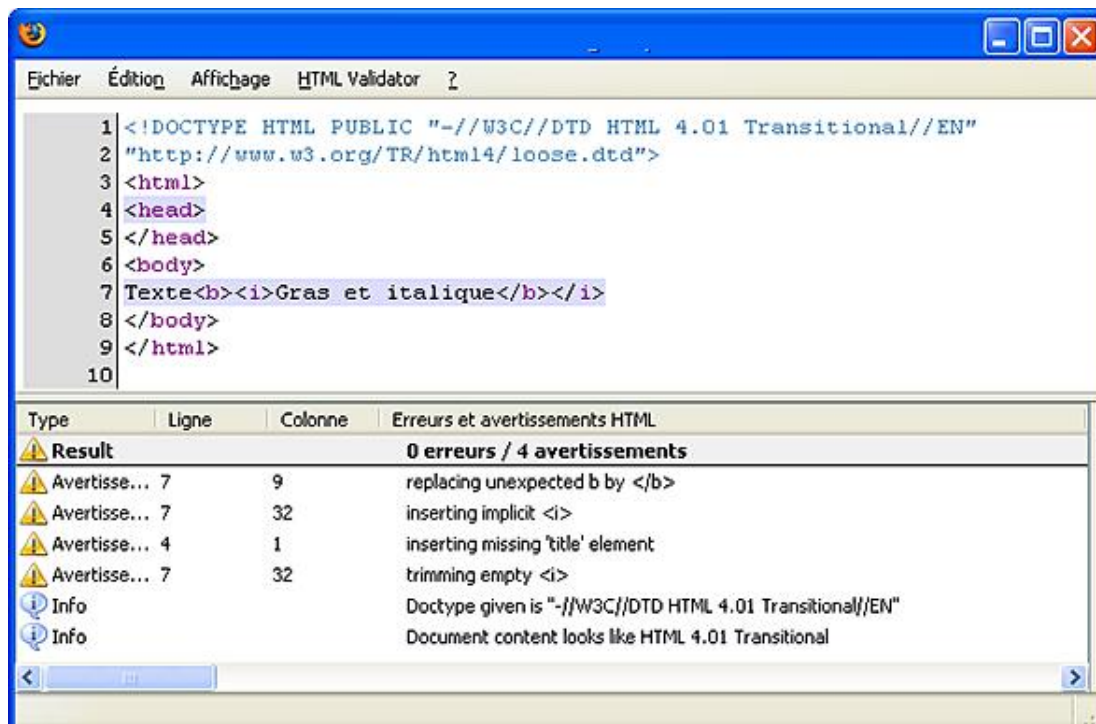
Html Validator (<http://extensions.geckozone.org/HTMLValidator>) est une extension pour Firefox basée sur la librairie Tidy initialement développée par le W3C. Ce validateur est présent sous la forme d'une icône située à droite de la barre de statut, et vérifie toutes les pages sur lesquelles vous naviguez.

Cette extension est déjà compatible pour Firefox 3.0.

L'exemple utilisé plus haut fournit les résultats suivants.



Après avoir double cliqué sur le petit triangle, la fenêtre de validation apparaît.



Le rôle des balises d'en-tête

Difficile de faire percevoir le rôle des balises d'en-tête, car lorsque vous composez votre page Html, elles n'ont AUCUNE importance sur l'affichage de votre site en local.

Par contre, une fois votre site en ligne, ces balises sont très prisées par les moteurs de recherche. Et quand on sait que quelques 60 % des pages de la toile sont accédées par ces moteurs de recherche, le rôle de ces balises devient primordial pour l'accessibilité et le nombre de visites de votre site.

Les moteurs de recherche, outre le contenu des balises meta, tiennent encore compte d'autres éléments comme le titre, les mots des premières lignes de la page ou la fréquence des mises à jour (cf. plus loin dans ce chapitre Conseils pour un bon référencement).

Chaque moteur de recherche a sa "formule magique" pour se différencier de ses concurrents et n'est donc pas très disposé à en divulguer tous les composants. Par contre, comme c'est souvent le cas sur Internet, les on-dit et autres rumeurs ne manquent pas.

Les balises meta indispensables

1. La balise de titre

Le titre de la page se retrouve entre la balise <title> ... </title>.

Exemple

```
<head>
<title>Le titre de la page</title>
</head>
```

Commentaires

La balise <title> est très importante car :

- Le titre est indiqué dans la barre de titre de la fenêtre lors de l'affichage par le navigateur.
- Le titre est affiché dans le navigateur dans la liste des pages déjà visitées.
- Le titre de la page constitue un élément **primordial** du référencement d'un site par les moteurs de recherche.

Il est vivement conseillé de mettre un titre (différent) sur **toutes** les pages d'un site ; même pour les pages qui n'apparaissent pas directement, comme les sous-pages d'une page de cadre.

Le fait d'intégrer un ou des mot(s) clé(s) dans le titre, en fonction du contenu de la page, est favorable pour la pertinence attribuée au site et donc pour un meilleur classement.

La longueur maximale de la balise <title> est en moyenne de 60 caractères (soit une dizaine de mots environ), mais ce nombre est différent suivant les moteurs de recherche.

2. La balise de description

Cette balise sert à donner une description de la page.

Exemple

```
<meta name="Description" content="une brève description de la page">
```

Commentaire

- Cette description est concise et attirante. Selon les moteurs de recherche seuls les 150 à 240 premiers mots seront repris.

3. La balise de mots clés

Cette balise contient une liste de mots clés.

Exemple

```
<meta name="Keywords" content="mot clé1,mot clé2,mot clé3,...">
```

Commentaires

- les mots clés doivent être séparés par une virgule ou des espaces.
- L'important consiste à trouver les bons mots clés relatifs au contenu de votre site, mais pas seulement ceux qui vous viennent à l'esprit, mais aussi les mots clés de vos lecteurs potentiels, qui n'ont peut-être qu'une connaissance sommaire du sujet. Quels mots, quels synonymes, quelles alternatives peuvent être utilisés pour décrire votre site ? Pensez aussi au pluriel et même aux fautes d'orthographe.

La tentation était grande de répéter un certain nombre de fois un même mot clé pour espérer un meilleur classement ; par exemple, la formulation suivante : `<meta name="Keywords" content="html,html,html, html,html">`. Désolé, mais cette astuce a vécu et est même maintenant pénalisée par les moteurs de recherche.



Les balises meta "Keywords" et "Description" ont longtemps servi au référencement d'un site web à destination des moteurs de recherche. L'emploi de ces balises a été très large, très intensif, peut-être même trop. C'est une des raisons pour lesquelles les moteurs de recherche en tiennent beaucoup moins compte aujourd'hui. Des moteurs, comme Google, Yahoo! Search ou MSN Search ne les prennent plus (ou presque plus) en compte.

On peut raisonnablement penser que les balises meta "Keywords" et "Description" ne sont plus aujourd'hui un critère de pertinence majeur des moteurs de recherche. Certains auteurs vont même jusqu'à les considérer comme inutiles à l'heure actuelle.

4. La balise "Robots"

La balise Robots permet de définir des règles à destination des robots pour leur indiquer la façon dont ils doivent indexer ou ne pas indexer la page ou le site.

La balise est :

```
<meta name="Robots" content="instructions pour les robots">
```

Les instructions sont :

Index Indique que vos pages peuvent être indexées par les robots.

NoIndex Indique aux robots de ne procéder à aucune indexation.

Follow Donne la permission aux robots de suivre les liens hypertextes des pages.

NoFollow Interdit aux robots de suivre les liens hypertextes du fichier.

All (défaut) Permet aux robots d'indexer vos pages et de suivre les liens hypertextes d'une page à l'autre.

None Indique aux robots de ne pas indexer vos pages et de ne pas suivre les liens.

Exemples

```
<meta name="Robots" content="None">
<meta name="Robots" content="Index,Nofollow">
```

Commentaires

- L'indication "Index,Follow" sont les valeurs par défaut de cette balise et n'a donc pas d'application pratique.
- Les indications index, noindex, follow et nofollow peuvent indifféremment être saisies en minuscules et en majuscules.

Les balises meta utiles

Les balises suivantes n'ont pas d'utilité pour le référencement mais constituent néanmoins une source d'informations appréciables pour l'utilisateur.

```
<meta name="Author" content="nom de l'auteur">
```

Cette balise est d'une utilité discutable car rares sont les moteurs de recherche qui en tiennent compte. Elle permet simplement de signer son œuvre.

```
<meta name="Copyright" content="Copyright © date nom">
```

Outre la valeur juridique et pratique, discutable, d'une mention de copyright sur le Web, cette balise permet au moins de respecter l'œuvre intellectuelle de l'auteur ! Citer vos sources est une simple question de courtoisie.

```
<meta name="Distribution" content="global, iu ou local">
```

Cette balise indique la destination de l'information de la page. Qu'elle soit "globale" et donc destinée à être largement diffusée, d'un usage interne (iu pour *Internal Use*) ou qu'elle soit "locale" et donc à diffusion restreinte.

```
<meta name="Generator" content="nom de l'éditeur Html utilisé">
```

Nullement utilisée par les moteurs de recherche ou par le navigateur, cette information intéresse cependant au plus haut point les responsables marketing des sociétés commercialisant des éditeurs Html pour calculer la part de marché de leur produit.

```
<meta name="Rating" content="Destination de votre audience">
```

Cette balise permet de définir l'audience du contenu de votre site. Les appréciations sont General, Mature, Restricted ou 14 years pour respectivement tout public, adulte, accès restreint ou 14 ans. Cet indice d'audience n'a aucune influence directe.

```
<meta http-equiv="Content-language" content="fr">
```

Cette balise déclare la langue utilisée dans le document Html. Elle est de plus en plus utile maintenant que les moteurs de recherche ont inclus la langue dans leurs critères de recherche.

```
<meta http-equiv="Reply-to" content="votre adresse e-mail">
```

Cette balise permet aux lecteurs avertis (ceux qui consultent le code source) de connaître votre adresse e-mail si elle n'apparaît pas sur la page qu'ils consultent. Il est peut-être plus utile de donner son adresse électronique que son nom.

```
<meta http-equiv="Reply-to" content="Adresse de votre page d'accueil">
```

Variante de la balise précédente. Cette balise permet de rappeler l'adresse de la page d'accueil de votre site ; ce qui peut être parfois utile pour ceux qui y sont entrés par une autre page que celle-ci.

Les balises meta éventuelles

1. Refresh

Un grand classique du Html (2.0 ?), même si cette balise meta n'est que rarement utilisée et alimente les trucs et astuces depuis des années. Cette balise meta charge automatiquement la page spécifiée à l'attribut URL après un délai de x secondes.

Exemple

```
<meta http-equiv = "Refresh" content="x;URL=adresse_de_la_page">
```



Cette balise est fréquemment utilisée pour rediriger automatiquement un visiteur dans le cas où l'adresse de votre site a été modifiée.

2. Transition

De très jolis effets de transition style Microsoft PowerPoint sont possibles avec simplement une ligne de code. Mais cela ne fonctionne que sous Microsoft Internet Explorer 4 et ultérieurs.

Exemple

```
<meta http-equiv="Page-Enter" content="revealTrans(Duration=1.0,
Transition=23)">
<meta http-equiv = "Page-Exit" content="revealTrans(Duration=1.0,
Transition=23)">
```

Commentaires

- Page-Enter et Page-Exit signifient que l'effet de transition se produira à l'entrée de la page ou à sa sortie.
- Duration détermine la durée de la transition en secondes. Elle est, dans l'exemple, de 1 seconde. À l'usage, cette durée ne se révèle pas d'une précision absolue.
- Transition est un nombre de 1 à 23 pour l'effet de transition choisi. Le chiffre 23 donne une transition aléatoire (au hasard). Les autres transitions se répartissent de 1 à 22. Ainsi, 7 ouvre la page de droite à gauche, 17 a le même effet mais en diagonale, 22 découvre la page avec un effet de lignes horizontales aléatoires, etc.
- Ces transitions ressemblent aux transitions de PowerPoint mais elles fonctionnent très bien même si PowerPoint n'est pas installé sur la machine de votre visiteur.
- Les effets de cette transition ne se voient qu'en entrant dans la page à partir d'une autre page (et donc pas en cliquant simplement sur le bouton **Actualiser**).



Ces effets de transition ont déjà été beaucoup utilisés. N'en n'abusez pas !

3. Expires

Cette balise meta indique au navigateur la date à laquelle la page doit être considérée comme périmée. Avec Firefox et Microsoft Explorer depuis sa version 5, une requête, pour un document dont la date est expirée initialisera une recherche par le navigateur sur le réseau au lieu de prendre la page éventuellement présente dans le cache de l'ordinateur. Ceci est très utile pour les pages fréquemment mises à jour.

Exemple

```
<meta http-equiv="Expires" content="Fri, 23 Feb 2007 10:49:02 GMT">
```

```
<meta http-equiv="Expires" content="0">
```

Commentaire

- Les robots de recherche peuvent retirer ces pages, déclarées périmées, de leur base de données.

4. Pragma

C'est une autre façon de contrôler le cache du navigateur. Avec cette balise meta, vous pouvez demander au navigateur de ne pas tenir la page dans le cache. Elle impose ainsi au navigateur de votre visiteur de recharger la page à chaque nouvelle visite et donc d'afficher les dernières mises à jour du site.

Exemple

```
<meta http-equiv="Pragma" content="no-cache">
```

Commentaire

- On rapporte dans les forums que cela fonctionne sous Firefox mais pas sous Internet Explorer qui préfère :

```
<meta http-equiv="Cache-control" content="no-cache">
```

Conseils pour un bon référencement

Hormis les balises meta, il existe des tas de trucs et astuces (plus ou moins vérifiables ou vérifiés) qui sont sensés améliorer le rang de votre site parmi les centaines de références renvoyées lors d'une recherche par un mot clé.

1. Référez-vous

Cela peut paraître une évidence mais il est inutile d'attendre qu'un robot de recherche vienne visiter votre site, vous risquez de devoir attendre longtemps... Google, le moteur de recherche le plus complet, estime lui-même que sa base de données ne reprendrait que 30 à 40% des sites du World Wide Web.

Alors préparez votre liste des moteurs de recherche, prévoyez quelques heures dans votre emploi du temps et au travail ! Sur la page d'accueil de chaque moteur de recherche, vous trouverez les options **Ajouter un site, Référencement** par exemple. En cliquant sur le lien, vous ouvrirez un formulaire, plus ou moins long, à remplir pour référencer directement votre site.

N'oubliez pas que certains moteurs de recherche style annuaire (comme Yahoo) se réservent le droit de reprendre ou de ne pas reprendre votre site, qu'il faut compter généralement 2 à 3 semaines avant d'apparaître dans la base de données et un bon mois avant que les internautes commencent à s'intéresser à votre site.

2. De l'importance des mots clés

Pour obtenir un bon classement, il faut non seulement définir ces mots clés dans la balise `<meta name="Keywords" content="mot-clé1,mot-clé2,mot-clé3,...">`, mais il est aussi recommandé :

- que les mots clés soient repris dans le premier paragraphe de la page.
- que les mots clés soient repris dans le titre de la page (la balise `<title>`).
- que les mots clés reviennent plus fréquemment que les autres mots de la page.

3. La page d'accueil avec des cadres

Les sites construits avec des cadres peuvent poser des problèmes pour le référencement par les moteurs de recherche (cf. Chapitre Les cadres - Les cadres et le référencement par Google).

Si un bon référencement est crucial pour le site (un site commercial par exemple), il est préférable d'éviter purement et simplement les cadres.

Néanmoins, sachant que les moteurs utilisent maintenant le contenu des balises `<noframes> ... </noframes>`, on y reconstruira une page d'accueil avec un petit texte descriptif du site, comportant un rappel des principaux mots clés et une liste des liens vers les autres pages du site.

4. La page d'accueil avec une image

Il est parfois joli de concevoir une page d'accueil composée d'une seule image sur laquelle le visiteur doit cliquer pour entrer dans le site.

Cette situation est assez inconfortable pour le robot à la recherche de mots clés car il n'a alors aucun texte (et donc pas de mots) pour faire son référencement... Il importe dans ce cas extrême de reprendre le mot clé dans l'attribut "alt" de la balise de l'image (cf. Chapitre Les images et arrière-plans - L'attribut alt).

5. La page d'accueil avec une image réactive

Les images réactives sont des images découpées en zones (cf. Chapitre Images réactives). Selon les zones cliquées par le lecteur, celui-ci est dirigé vers une page Html déterminée.

Il est assez risqué, pour obtenir un bon référencement, de prévoir une page d'accueil qui ne comporterait qu'une image réactive comme seul outil de navigation pour votre site. En effet, beaucoup de moteurs de recherche ne

poursuivent pas leur indexation dans les fichiers d'une image réactive. Il est donc préférable de la réserver à d'autres usages à l'intérieur de votre site.

6. Et encore

Sont ou seraient bénéfiques pour un bon référencement :

- une adresse Web reprenant le titre du site ou des mots clés du site.
- un contenu de la page reprenant les différents mots clés, si possible dans les premières lignes du texte.
- une structuration de la page avec des titres selon la balise `<hx> ... </hx>`.
- des liens vers des pages du site sur des mots clés. Le nom du fichier cible reprendra également le mot clé concerné.
- un attribut alt sur les images.
- un menu de navigation intuitif où l'on évitera le JavaScript.
- des liens externes vers des sites pertinents, bien classés par les moteurs de recherche.
- des liens d'autres sites, également bien classés par les moteurs de recherche, vers le vôtre.
- des mises à jour fréquentes.
- ...

7. Désolé...

Les astuces suivantes pour dissimuler des mots clés, ne fonctionnent plus et sont même pénalisées par les moteurs de recherche :

- des mots clés invisibles dans la même couleur que le fond de la page.
- des mots clés repris en commentaire (balises `<!-- ... -->`).
- les mêmes mots clés repris indéfiniment dans la balise `<meta name="Keywords">`.
- les mots clés dans un élément de formulaire `<input type="hidden"...>` (éléments cachés - cf. chapitre Les formulaires - Les formulaires cachés (hidden)).

Exemple de balises d'en-tête

Appliquées à un site reprenant cet ouvrage, les balises d'en-tête pourraient être :

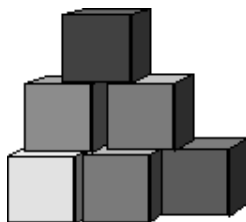
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
"http://www.w3.org/TR/REC-html40/loose.dtd">
<html>
<head>
<title>Le code source HTML</title>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<meta http-equiv="Content-language" content="fr">
<meta name="Description" content="Le code source des pages Web
expliqué de façon pédagogique du HTML au XHTML.">
<meta name="Keywords" LANG="fr" content="HTML,html,Html,HTML 2.0, HTML 3.2,
HTML 4.0, XHTML,XHTML 1.0,JavaScript,DHTML,CSS,feuilles de style">
<meta name="Robots" content="Index,Follow">
<meta name="Copyright" content="Copyright © ENI Editions">
<meta name="Author" LANG="fr" content="Van Lancker Luc">
<meta name="Identifiant-URL" content="http://www.editions-eni.com/">
<meta http-equiv="Reply-to" content="editions@ediENI.com">
<meta name="Revisit-after" content="14 days">
</head>
<body>
. . . .
</body>
</html>
```

Le concept d'images à zones réactives

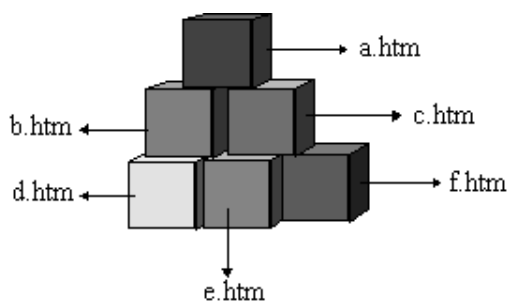
Les images réactives sont abordées dans un chapitre distinct du chapitre Les images et arrière-plans, consacré aux images (normales), car cette matière, d'un usage somme tout limité, réclame un niveau d'abstraction et de connaissance du Html assez aigu.

Les images réactives permettent d'effectuer des liens distincts en fonction de parties d'images ou de zones [area] prédéfinies de celles-ci. Cette particularité peut se révéler fort utile pour définir, par exemple, des outils de navigation à l'intérieur de votre site.

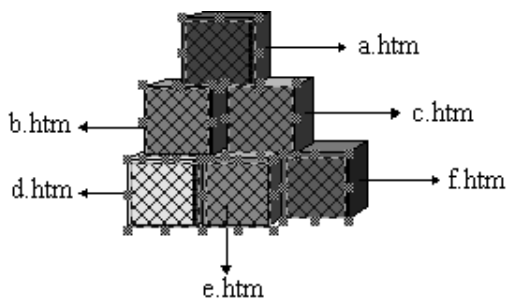
Soit une image,



pour chaque zone retenue (ici un cube), un fichier est associé.



Pour ce faire, des zones dans l'image sont définies, un peu comme avec une carte de géographie [map en anglais] et à chacune de ces zones, est associé un fichier.



Une image réactive est ainsi obtenue (ou image "mapée") car elle est découpée en zones à l'instar des cartes géographiques. Les balises Html qui reprennent à la fois les zones de l'image et les fichiers associés sont appelés carte (ou map).



Il existe plusieurs méthodes pour réaliser cet effet d'images réactives : les méthodes NCSA, CERN et CSIM. La méthode CSIM (*Client Side Image Map*) est de loin la plus utilisée à l'heure actuelle car elle fait partie, à part entière, du langage Html. Avec cette méthode, les fichiers de la carte (map) sont inclus dans la page Html et ne nécessitent pas l'appel à des ressources additionnelles du serveur pour réaliser leurs fonctions. C'est cette méthode qui sera utilisée dans cet ouvrage.

Les balises expliquées

Les balises des images réactives peuvent sembler très bizarres ; cependant, une fois décodées, elles déterminent (en peu d'éléments) tout ce dont le navigateur a besoin pour les traiter.

1. La balise de l'image réactive

```

```

pour une carte dans le même fichier.

En fait, il suffit d'ajouter à la balise classique de l'image, l'attribut `usemap` pour avertir le navigateur qu'il doit employer pour celle-ci une carte (map) et le nom de la carte en question.



Remarquez le système du dièse #, propre aux ancres (cf. Chapitre Les liens - Les liens à l'intérieur d'une page).

2. Les balises de la carte

Pour rappel, la carte (map) est découpée en zones (area).

Exemple

```
<map name="nom_de_la_carte">
<area shape="forme" coords="coordonnées" href="destination"
alt="commentaires" target="nom_de_cadre">
... autres balises area ...
</map>
```

Soient les éléments suivants :

a. La balise `<map name="nom_de_la_carte">`

Un nom est attribué à la carte et c'est ce nom qui sera retenu dans l'attribut `usemap`.

b. L'attribut `shape`

L'attribut `shape="forme"` détermine la forme de la zone :

`rect` pour un rectangle.

`circle` pour un cercle.

`polygon` pour un polygone irrégulier.

c. L'attribut `coords`

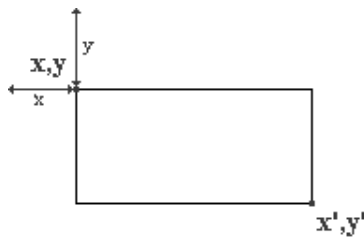
L'attribut `coords="coordonnées"` note les coordonnées qui permettront au navigateur de reconstituer la forme géométrique.

La véritable difficulté pour le webmestre est de trouver les coordonnées des points dans l'image. C'est ici qu'une application de traitement de l'image comme Adobe Photoshop ou Paint Shop Pro ou Gimp, se révèle utile.

Nous détaillons ci-après la façon spécifique à chaque méthode, de noter ces coordonnées.

Pour un rectangle :

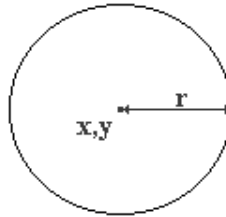
```
coords="x,y,x',y' "
```



Pour un cercle :

`coords="x,y,r"`

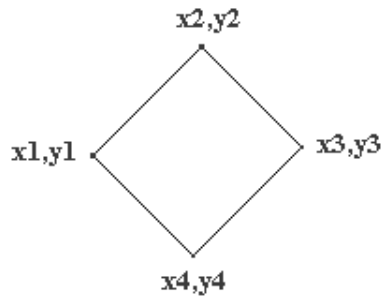
soit le point central (x,y) et le rayon (r).



Pour un polygone :

`coords="x1,y1,x2,y2,x3,y3,x4,y4"`

et ce pour autant de points qu'il y a dans le polygone.



d. L'attribut link

L'attribut `link="destination"` spécifie le fichier associé à la zone sélectionnée.

L'adressage se fait de façon tout à fait classique en Html.

e. L'attribut alt

L'attribut `alt="commentaires"` permet d'ajouter un commentaire.

Ce commentaire, tout comme l'attribut `alt` des images, sera affiché sous forme d'une info-bulle.

f. L'attribut targets (facultatif)

L'attribut `target="cadre"` permet de spécifier la fenêtre de cadre dans laquelle doit s'ouvrir le fichier spécifié. Ceci nécessite, bien entendu, une connaissance pointue des cadres (cf. Chapitre Les cadres).

Exemple

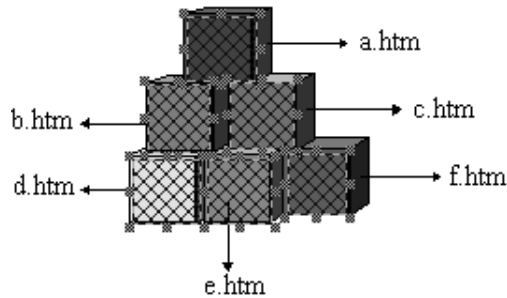
Soit le fichier Html suivant :

```
<html>
<body>
<p align="center">
```



```

</p>
</body>
</html>
```



Et la carte nommée "cartons" de l'image "cubes.gif" :

```
<map name="cartons">
<area shape=rect coords="37,9,72,40" href="a.htm">
<area shape=rect coords="18,46,47,79" href="b.htm">
<area shape=rect coords="61,43,93,78" href="c.htm">
<area shape=rect coords="9,84,36,119" href="d.htm">
<area shape=rect coords="48,85,77,116" href="e.htm">
<area shape=rect coords="89,81,123,115" href="f.htm">
</map>
```

L'insertion de la carte (map) dans le fichier Html se réalise en deux temps :

1. Ajout de l'information, pour le navigateur, usemap="#cartons" dans la balise de l'image.
2. Insertion de la carte n'importe où dans le fichier Html.

Le fichier final devient :

```
<html>
<body>
<p align="center">

</p>
<map name="cartons">
<area shape=rect coords="37,9,72,40" href="a.htm">
<area shape=rect coords="18,46,47,79" href="b.htm">
<area shape=rect coords="61,43,93,78" href="c.htm">
<area shape=rect coords="9,84,36,119" href="d.htm">
<area shape=rect coords="48,85,77,116" href="e.htm">
<area shape=rect coords="89,81,123,115" href="f.htm">
</map>
</body>
</html>
```

Les logiciels disponibles

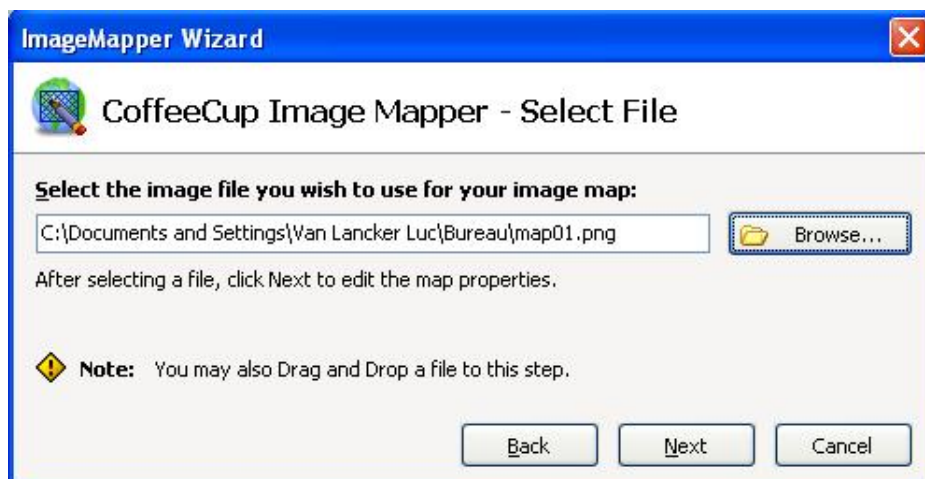
Les éditeurs Html évolués, comme par exemple Dreamweaver, possèdent des fonctions pour réaliser directement des images réactives.

Pour ceux qui préfèrent encoder manuellement, il existe quelques applications qui peuvent aider à réaliser les images réactives. Citons MapThis que vous pouvez télécharger à l'adresse www.lehtml.com/htmlplus/mapthis.htm. Ce logiciel est freeware, en anglais, efficace mais dont le look devient désuet. Retenons CoffeeCup Image Mapper (www.coffeecup.com/image-mapper/), un shareware, en anglais mais assez intuitif.

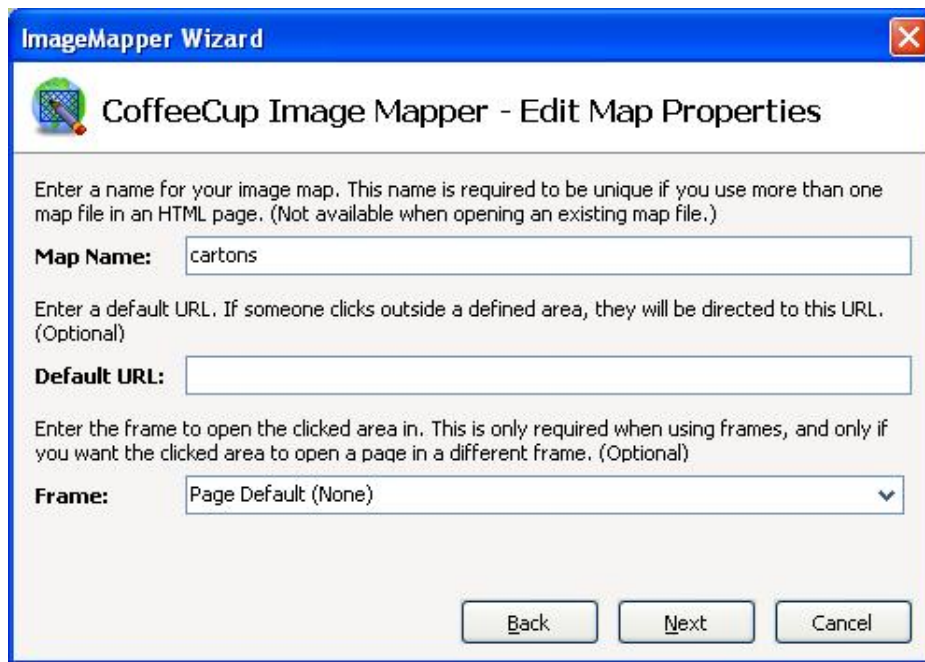
- Après l'avoir téléchargé et installé, ouvrez l'application. L'assistant (disponible aussi par **File - Map Wizard**) vous propose de créer ou d'ouvrir une image mapée enregistrée.
- Retenez l'option **Create a New Image Map**. Puis cliquez **Next**.



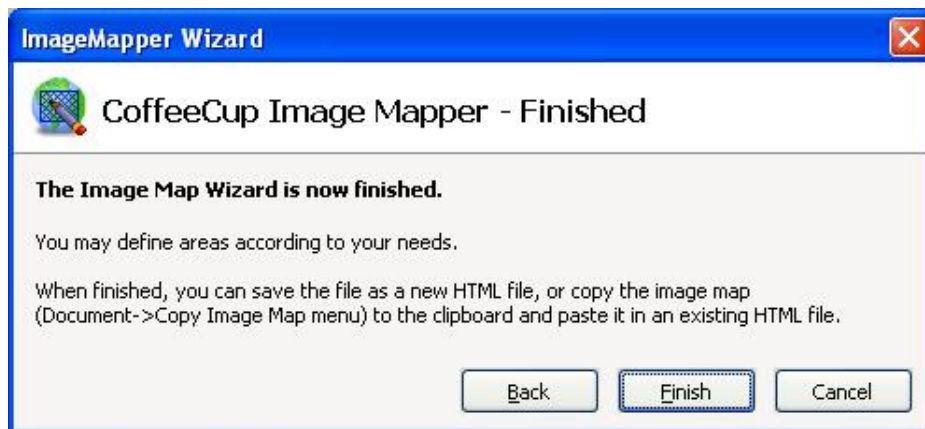
- L'écran suivant demande de désigner le fichier image. Une fois l'opération effectuée, cliquez **Next**.



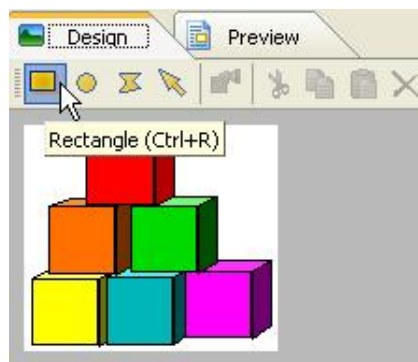
- Dans le troisième écran, il faut donner un nom à la map (*Map Name*) qui sera créée. Pour respecter le code élaboré au point 2, nous l'appellerons "cartons". Ensuite cliquez **Next**.



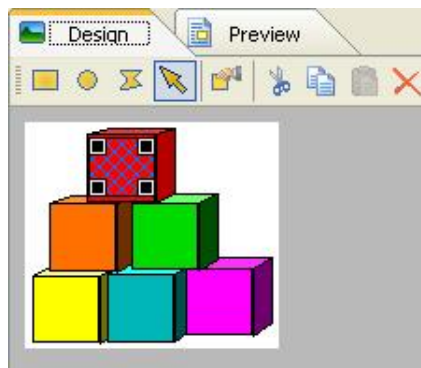
- La dernière fenêtre vous informe simplement que vous pouvez définir maintenant les zones de l'image réactive. Cliquez **Finish**.



- Choisissez l'objet rectangle dans la barre d'outils.



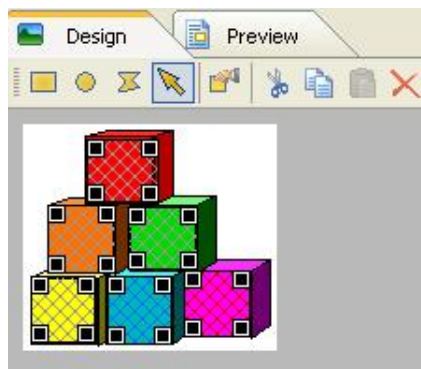
- Sélectionnez le cube supérieur comme dans la capture d'écran.



- À l'extrémité supérieure droite de la fenêtre se trouve la boîte de propriétés de la zone (*Area Properties*). Encodrez l'URL du fichier associé à la zone et le texte de l'attribut Alt (*Alt Text*).

Area Properties	
Name	
URL	a.htm
Alt Text	vers a
Target	<input type="button" value="v"/>
MouseOver Text	
Shape	Rectangle

- Procédez de même pour toutes les autres zones de l'image.



- Vous trouvez le code généré que vous pouvez copier/coller dans la partie inférieure gauche.

```
<!-- Beginning of Client Side Image Map -->

<map name="cartons">
  <area shape="rect" coords="13,43,50,80" href="b.htm" alt="vers b">
  <area shape="rect" coords="56,44,94,78" href="c.htm" alt="vers c">
  <area shape="rect" coords="4,81,39,117" href="d.htm" alt="vers d">
  <area shape="rect" coords="34,8,73,42" href="a.htm" alt="vers a">
  <area shape="rect" coords="45,81,80,118" href="e.htm" alt="vers e">
  <area shape="rect" coords="86,78,122,114" href="f.htm" alt="vers f">
</map>
<!-- End of Client Side Image Map -->
```

Introduction

Les feuilles de style sont des ajouts aux balises Html qui prennent en charge la présentation et l'aspect visuel des pages Web.

Les feuilles de style sont reconnues par Internet Explorer depuis la version 4. Cependant, Microsoft les applique parfois d'une façon assez personnelle et pas toujours conforme aux spécifications du W3C. Ces interprétations, voire ces erreurs, perdurent jusqu'à la version 7. La version 8 qui est annoncée à ce jour (début 2009) devrait enfin corriger ces quelques bugs.

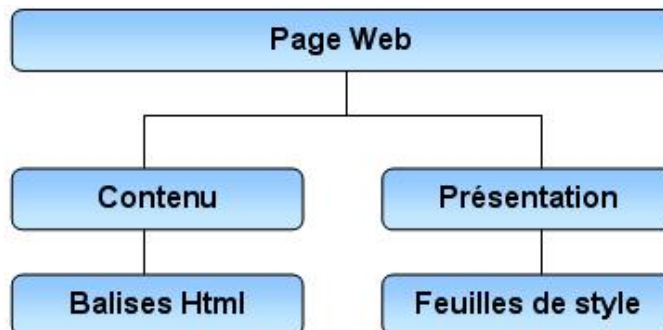
Firefox par contre, s'est fixé comme objectif de respecter scrupuleusement les spécifications du W3C concernant les feuilles de style.

Nous ne manquerons pas de mettre en avant quelques différences d'interprétation entre Internet Explorer et Firefox.

L'objectif de départ

Introduites par Microsoft avec Internet Explorer 3, les feuilles de style en cascade, appelées aussi CSS (*Cascading Style Sheets*), ont fait petit à petit leur apparition dans le paysage de la publication de pages Web. Avec la version Html 4.0 de 1999, elles ont même été dûment reconnues et fortement conseillées par le W3C. Force est de constater que le concept prend de plus en plus d'importance et que les sites de sociétés professionnelles ou de grands organismes l'adoptent largement.

L'essence des feuilles de style (qui ne sont pas sans rappeler les feuilles de style de Microsoft Word) est de séparer le contenu du document de tout l'aspect présentation de celui-ci.



Les avantages en sont multiples :

- pour les sites importants, où plusieurs intervenants peuvent participer, chacun peut apporter son contenu textuel en respectant les critères de présentation prédéfinis ; ce qui, malgré la disparité des intervenants, assure une unité et une cohérence dans l'aspect graphique du site.
- en cas de "relookage" du site, il suffit de modifier quelques propriétés de style pour que les modifications s'appliquent à l'ensemble des pages d'un site.

Un petit exemple pratique :

Vous souhaitez que tous les chapitres abordés dans votre site aient la même taille, la même police, la même couleur et le même formatage gras. Vous devriez alors encoder à chaque chapitre, la série de balises suivantes.

```
<font face="Arial"><font size="5"><font color="#999999"><b>Chapitre A :  
Objectif de départ</b></font></font></font>
```

Il est bien plus simple de définir une fois pour toutes, un style que vous appellerez par exemple "niveau1" et qui reprendra les mêmes caractéristiques. En cours de travail, il vous suffira de spécifier que ce qui suit est du "niveau1".

L'utilisation pratique

Le Html n'est pas très riche en possibilités de présentation. Les feuilles de style, en repartant d'une feuille blanche, apportent de nombreuses variations par rapport aux thèmes connus du Html ; comme par exemple un texte plus grand que la balise `<h1>`, un interligne différent, un formulaire de couleur, etc.

Ainsi, le concepteur amateur débutant ou confirmé, profite souvent des effets des feuilles de style pour agrémenter les pages Html. Les feuilles de styles ne sont alors plus utilisées à longueur de page mais employées de façon ponctuelle, souvent directement dans le code source, à l'image de scripts ou autres animations.

La syntaxe d'un élément de style

La définition générale d'un élément de style est la suivante :

```
balise { propriété de style: valeur; propriété de style: valeur }
```

Ce qui se lit : à la balise déterminée, appliquer la première propriété de style ainsi que la propriété de style suivante.

Exemple

```
h2 {font family: Arial; color: #999999}
```

Ce qui se lit : à la balise de titre h2, appliquer la propriété de police de valeur Arial, et la couleur désignée.

Commentaires

- Les feuilles de style s'ajoutent principalement à des balises, par exemple p, body, mais aussi à des éléments a:link pour un lien (non-visité).
- Les propriétés de style sont entourées par des accolades "{".
- Le couple "propriété de style et valeur" est séparé par un double point (:).
- Chaque couple "propriété de style et valeur" est séparé par un point-virgule (;).
- Il n'y a pas de limite pour le nombre de couples "propriétés de style et valeur".
- L'espace entre propriétés de style et valeur n'est pas obligatoire mais aide fortement à la lisibilité du code source.
- Bien que les styles ne soient pas sensibles aux majuscules et minuscules (case insensitive), l'usage veut que les feuilles de style soient notées en minuscules.
- Pour la lisibilité toujours, les styles peuvent être écrits sur plusieurs lignes :

```
h2 {font family: Arial;
    color: #999999}
```

- Plusieurs valeurs peuvent être attribuées à une même propriété. Dans ce cas, les différentes valeurs sont séparées par des virgules.

```
h3 {font-family: Arial, Helvetica, sans-serif}
```

- Un même style peut être attribué à plusieurs balises (séparées par des virgules).

```
h1, h2, h3 {font-family: Arial; font-style: italic}
```

➤ Alors que le Html fonctionne avec des abréviations par exemple b pour bold (gras), les feuilles de style sont plus "bavardes" mais aussi plus explicites. Ainsi pour mettre en gras, le style équivalent à la balise est font-weight:bold. Cela a l'avantage, avec quelques notions d'anglais ou la force de l'habitude, de pouvoir "lire" assez aisément les feuilles de style.

Les feuilles de style internes

Les feuilles de style dites internes s'appliquent à tout un document Html mais à ce seul document.

Il est ainsi possible d'incorporer les feuilles de style internes dans l'en-tête, entre les balises <head>.

Exemple

```
<head>
<style type="text/css">
La ou les feuille(s) de style
</style>
</head>
```

Commentaires

- La balise <style> ... </style> informe le navigateur que le contenu comporte des feuilles de style.
- Le type de la définition de format type="text/css" est facultatif dans la pratique.

Exemple

Un grand classique est l'effet de style qui enlève le soulignement (la "décoration") à **tous** les liens d'un document Html. Cet effet s'obtient par le code :

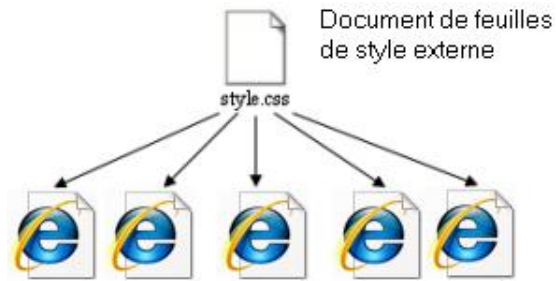
```
<html>
<head>
<style>
<style type="text/css">
a {text-decoration:none}
</style>
</head>
<body>
<a href="test.htm">Ceci est bien un lien</a>
</body>
</html>
```

Résultat



Les feuilles de style externes

Les feuilles de style externes (externes car regroupées dans un fichier séparé), sont idéales pour donner un style commun à toute une série de pages. Plutôt que de devoir mettre dans chaque page les lignes de code du style, il suffit de les encoder une seule fois et elles pourront servir à toutes les pages qui y font référence.



Ceci se réalise en deux étapes :

Il convient d'abord de créer avec un éditeur de texte simple, genre Bloc-notes de Windows, un fichier qui ne contient rien d'autre que les feuilles de style (propriété et valeur) enregistré sous l'extension réservée aux feuilles de style soit .css. Notre fichier est ici nommé style.css.

Il faut ensuite dans chaque page Web appeler ce document de feuilles de style externe ; ce qui se fait par la balise `<link>` qui se positionne dans l'en-tête du document Html `<head>`.

Exemple

```
<head>
<link rel=stylesheet type="text/css" href="style.css">
</head>
```

Commentaires

- La balise `<link>` avertit le navigateur qu'il faudra réaliser un lien.
- L'attribut `rel=stylesheet` précise qu'il y trouvera une feuille de style externe. Remarquez que c'est *stylesheet* au singulier donc sans s.
- L'attribut `type="text/css"` précise que l'information est du texte et du genre cascading style sheets (css).
- L'attribut classique de lien `href="destination"` donne le chemin d'accès et le nom du fichier à lier. Ici le fichier style.css est supposé être dans le même dossier que les pages Web.

L'application d'un style dans une page Html

Il est aussi possible d'appliquer un effet de style en local, soit ponctuellement à une balise déterminée. Cette façon de faire peut paraître illogique et peu conforme à l'esprit des feuilles de style qui est de définir un style déterminé valable pour la globalité du document. Mais elle permet d'ajouter très simplement un effet visuel à la page.

Le style s'applique alors à une balise déterminée du corps du document.

Reprenez l'exemple des feuilles de style internes et supprimez le soulignement **d'un** lien spécifique.

Le code devient :

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<a style="text-decoration:none" href="test.htm">Ceci est bien
un lien</a>
</body>
</html>
```

➤ Vous pouvez avoir une feuille de style externe, des spécifications de style internes et des styles appliqués en local dans la page. C'est pour cette raison que l'on parle de feuilles de style... en cascade (*cascading*). Le navigateur accorde la priorité au style le plus proche et applique celui-ci.

Les éléments class et id

Une feuille de style est appliquée à une balise donnée. Les éléments `class` et `id` permettent, eux, de donner des effets de style différents à une même balise.

1. L'élément class

La définition d'un style était :

```
balise { propriété de style: valeur }
```

Elle devient :

```
balise.nom_de_classe { propriété de style: valeur }
```

Comme la mention de la balise est facultative, on peut aussi indiquer :

```
.nom_de_classe { propriété de style: valeur }
```

➤ Remarquez le point entre balise et nom_de_classe ainsi que l'emploi du point devant nom_de_classe.

Pour appeler l'effet de style dans le document, ajoutez le nom de la classe à la balise :

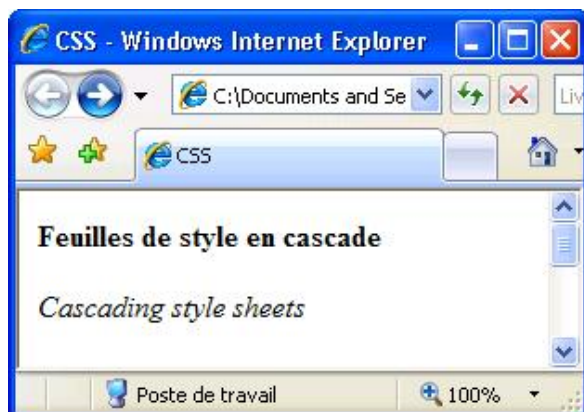
```
<balise class="nom_de_classe"> .... </balise>.
```

Exemple

Mettez un type de paragraphe en gras et en rouge désigné par le nom "important" et un autre type de paragraphe en italique nommé "traduction".

```
<html>
<head>
<title>CSS</title>
<style type="text/css">
.important {font-weight: bold; font-color: red}
.traduction {font-style:italic}
</style>
</head>
<body>
<p class="important">Feuilles de style en cascade</p>
<p class="traduction">Cascading style sheets</p>
</body>
</html>
```

Résultat



2. L'élément id

Il existe aussi l'élément `id` dont la fonction est similaire à l'élément `class` à ceci près. Comme la convention `nom/point/nom` est utilisée aussi en JavaScript (cf. Chapitre Les autres langages du web), il a fallu trouver une autre convention d'écriture pour utiliser les feuilles de style avec du JavaScript.

Ce sont les `id`, aussi appelés les identifiants. Les `id` fonctionnent exactement comme les `class`.

La syntaxe est :

```
#nom_de_id { propriété de style: valeur }
```

Et pour l'appeler :

```
<balise id="nom_de_id"> .... </balise>
```

Un seul appel à `#nom_de_id` par page pourra être effectué.

En conclusion, si vous pensez utiliser des feuilles de style, mais sans utiliser des scripts, oubliez au plus vite `id` et utilisez exclusivement les éléments `class`.

Par contre, lorsque vous souhaitez utiliser des scripts avec les feuilles de style, la notion de `id` vous sera alors indispensable.

Les balises et <div>

Voilà des balises que l'on attendait et que l'on ne finit pas de découvrir. Ces balises introduisent dans la conception de pages Web ce que l'on pourrait comparer aux zones de texte pour ceux qui utilisent les outils bureautiques ou aux calques pour les infographistes.

Ces zones ou boîtes ainsi déterminées ne prennent de sens que par les effets de style qui leur sont appliquées. Ces zones peuvent s'inclure dans le texte de la page mais aussi en relief par rapport à la page grâce à la propriété *z-index* des feuilles de style.

La balise <div> crée une boîte ou un élément de type bloc. Ces éléments servent à distinguer des parties entières de texte comme les paragraphes <p>, les titres <h>, les listes ou , les citations <blockquote>, etc. Ces éléments sont distingués du reste du contenu par un retour chariot.

La balise est un élément en-ligne qui sert à distinguer une portion de texte incluse dans un élément bloc. Par exemple, quelques mots d'un paragraphe.

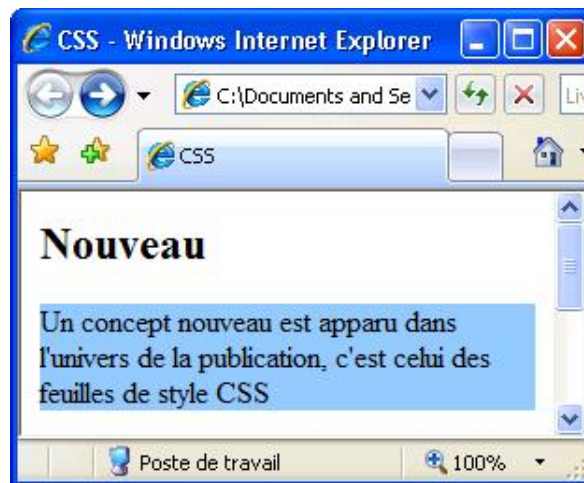
La syntaxe est très simple. Il suffit d'entourer la zone sélectionnée par ... ou <div> ... </div>. Associée à un style, la syntaxe complète est donc :

```
<span style="propriétés: valeur"> ... </span>
<div style="propriétés: valeur"> ... </div>
<span class="nom_de_classe"> ... </span>
<div class="nom_de_classe"> ... </div>
```

Exemple de <div>

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<h2>Nouveau</h2>
<div style="background-color:#99ccff">Un concept nouveau est apparu
dans l'univers de la publication, c'est celui des feuilles de style CSS
</div>
</body>
</html>
```

Résultat



Exemple de

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<p>Un concept nouveau est apparu dans l'univers de la publication, c'est
celui des <span style="background-color:#99ccFF;" feuilles de style
CSS</span>.
```

```
</p>
</body>
</html>
```

Résultat



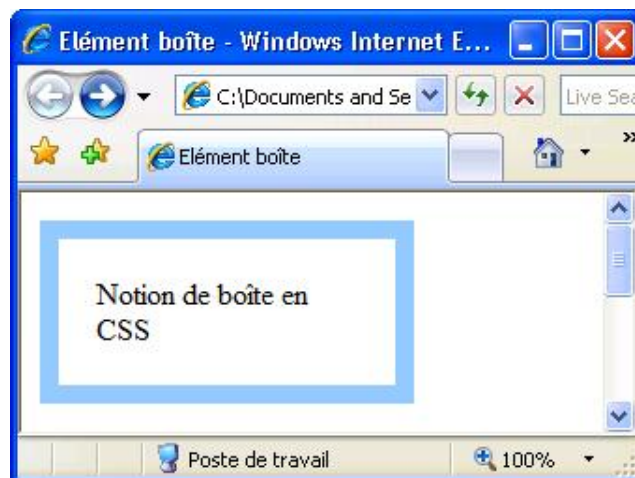
Ce concept de boîte est dans certaines circonstances (avec ou sans doctype) interprété différemment par Explorer 6 ou 7 et par Firefox. Sans entrer dans le détail, ce cas fournit un bel exemple de différences d'affichage d'un même code par des navigateurs différents.

Pour de plus amples explications, vous pouvez vous reporter utilement à l'ouvrage "Des CSS au DHTML" de la collection Ressources Informatiques au chapitre 8.

Soit le code :

```
<html>
<head>
<title>Elément boîte</title>
<style type="text/css">
<style>
<!--
p {width: 200px;
    border: 10px solid #99ccff;
    padding: 20px;}
</style>
</head>
<body>
<p>Notion de boîte en CSS</p>
</body>
</html>
```

Internet Explorer 7 affichera :



Tandis que Firefox quant à lui affichera :

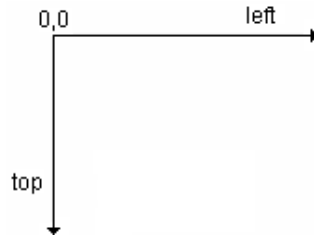


La différence est sensible et peut modifier complètement l'aspect d'une page Web.

Le positionnement avec et <div>

Il est désormais possible de positionner, au pixel près, du texte ou une image avec les feuilles de style. Le positionnement peut se réaliser en position absolue ou en position relative.

La position absolue {position: absolute} se détermine par rapport au coin supérieur gauche de la fenêtre du navigateur. Les coordonnées du point initial sont top = 0 et left = 0. Les coordonnées d'un point s'expriment en pixels, de haut en bas pour top et de gauche à droite pour left.



La position relative {position: relative} se détermine par rapport à la position normale ou originale de l'élément dans la page, par exemple un élément du code Html.

Le positionnement absolu sera le plus souvent utilisé car il est plus facile et plus sûr. Mais il est important de ne pas oublier que le visiteur de votre site n'a peut-être pas la même résolution d'écran que vous. Ce qui tombe pile avec une résolution 800x600, risque de se retrouver n'importe où avec une résolution 1024x768.

À titre d'exemple, créez un effet d'ombrage en superposant, mais de façon légèrement décalée, un texte en rouge et son ombre (le même texte) en gris.

- Pour ce faire, créez deux zones en position absolue sous forme de de 200 pixels de large et de 55 pixels de haut.

Le code en est :

```
<span style="position:absolute; width:200px; height:55px;
```

- Positionnez le texte en rouge (red) à 40 pixels à gauche et 23 pixels du bord supérieur (top).

Le code d'une des balises se complète :

```
... color:red; left: 40px; top: 23px">
```

Le texte en gris (gray) est légèrement décalé soit un peu plus à gauche et un peu plus à droite.

L'autre balise devient :

```
... color:gray; left: 47px; top: 31px">
```

Le code complet est alors :

Exemple

```
<html>
<head>
<title>CSS</title>
</head>
<body>
<span style="position:absolute; width:200px; height:55px; color:gray;
left: 47px; top: 31px">
<font size="7"><b>STYLES</b></font>
</span>
<span style="position:absolute; width:200px; height:55px; color:red;
left: 40px; top: 23px">
<font size="7"><b>STYLES</b></font>
</span>
</body>
</html>
```



Les unités de mesure

Le Html n'utilisait que le pixel ou le pourcentage. Les feuilles de style permettent d'utiliser d'autres unités de mesure :

px	en pixel
%	en pourcentage
in	en pouce (1 inch = 2,54 centimètres)
cm	en centimètre
mm	en millimètre
pt	en point (1 point = 1/72 inches)
pc	en pica (1 pica = 12 points)

La codification des couleurs

Outre la codification par nom et en valeur hexadécimale, les feuilles de style introduisent d'autres façons d'encoder les couleurs. Soit :

- par un nom fonctionnel,
- par la valeur hexadécimale composée de 6 chiffres précédée d'un dièse # soit #000000. Donc comme en Html mais sans les guillemets !
- par une valeur hexadécimale à 3 chiffres. Chaque chiffre est alors implicitement dupliqué, ainsi #fd3 est équivalent à #ffdd33,
- par la notation fonctionnelle rgb qui prend 3 arguments en l'occurrence 3 nombres entiers compris entre 0 et 255 ou 3 pourcentages entre 0% et 100%, par exemple, color : rgb(255,0,0) ou color : rgb(50%,50%,50%).